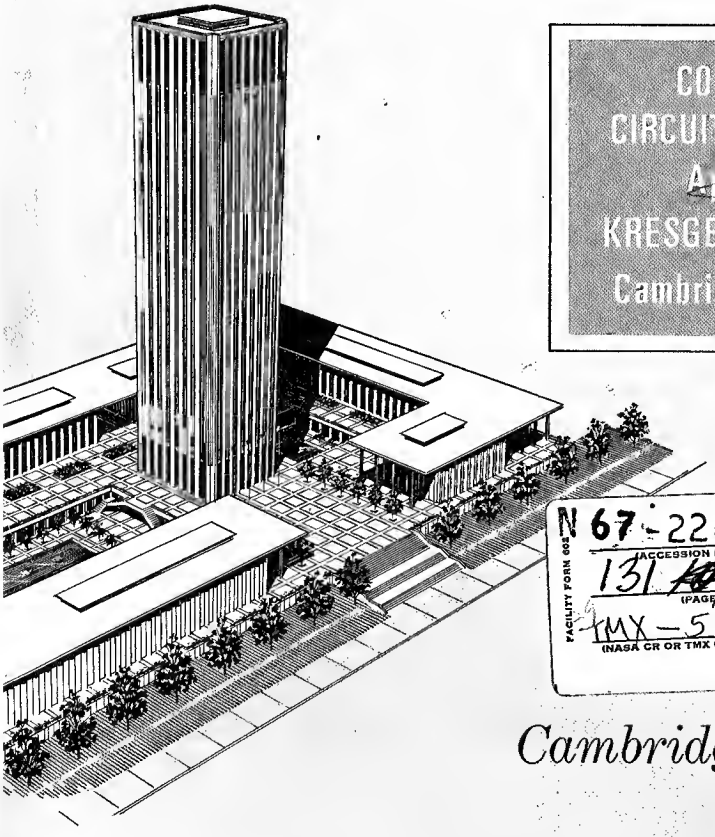
ELECTRONICS RESEARCH CENTER



COMPUTER-AIDED
CIRCUIT DESIGN SEMINAR
April 11-12, 1967
KRESGE AUDITORIUM, M.I.T.
Cambridge, Massachusetts

Cambridge, Massachusetts

COMPUTER-AIDED CIRCUIT DESIGN SEMINAR

KRESGE AUDITORIUM
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Cambridge, Massachusetts

April 11-12, 1967

Sponsored by
NATIONAL AERONAUTICS
AND SPACE ADMINISTRATION
ELECTRONICS RESEARCH CENTER
Cambridge, Massachusetts

FOREWORD

Man's attempts to cope with the increasing complexity of our present society have been more and more dependent upon electronics. Through sophisticated computing machines, more efficient transmission methods and automation, electronics is providing logical extensions of man's innate capability to process and control information. Implicit in these expanding relationships is the need for more efficient communication between man and machine.

Computer-aided circuit design -- the topic of this seminar -- is representative of such a logical extension. The variety of newly developed processing techniques capable of providing large-scale, complex electronic systems necessitates even higher levels of sophistication in systems organization and logical design if the opportunities for improvements in reliability, size, maintainability, and cost are to be realized.

We are most appreciative of the favorable response on the part of the individual contributors to this seminar, each of whom was selected on the basis of important contributions to this aspect of computer technology. Summaries of the presentation material have been compiled in this document.

We are grateful to all those present, both for indication of interest in the proceedings and for the evidence of active participation in this growing field.

Robert L. Trent
General Chairman

PANEL DISCUSSION

The Seminar will conclude with a panel discussion on "Time Sharing versus Batch Processing."

Panel members include:

Theodore Bashkow
Columbia University, New York, N. Y.

Joseph Braddock
Braddock, Dunn and McDonald, El Paso, Tex.

Gerald Cohen
University of Rochester, Rochester, N. Y.

Michael L. Dertouzos
M. I. T. Department of Electrical Engineering
Cambridge, Mass.

James Dobbie
General Electric Computer Division, Phoenix, Ariz.

William Happ
NASA Electronics Research Center
Cambridge, Mass.

CONTENTS

CONTENTS

COMPUTER METHODS OF NETWORK ANALYSIS

By

DR. FRANKLIN H. BRANIN, JR.

Dr. Branin received his B.A. degree in Chemistry from Wesleyan University in 1942. After several years as a radar-electronics officer in the Navy, he returned to graduate school and received his M.A. and Ph.D. degrees in Physical Chemistry from Princeton University in 1948 and 1950. From 1948 to 1950, he also worked at RCA laboratories on transistor technology and was recalled to active duty at the Naval Ordnance Laboratory, White Oak, Maryland during 1951. From 1952 to 1957, he worked first at Los Alamos Scientific Laboratory and then at Shell Development Company, Emeryville, California, on electronic instrumentation.

Since 1957, he has worked at IBM, Poughkeepsie, New York, mainly on computer methods of network analysis. His contributions to this field include papers on Kron's method, a network approach to structural analysis, a new method of A-C analysis, and a network-topological description of the vector calculus and Maxwell's equations.

Dr. Branin is a member of Phi Beta Kappa, Sigma Xi, RESA, and a senior member of IEEE.

COMPUTER METHODS OF NETWORK ANALYSIS

Franklin H. Branin, Jr.
International Business Machines Corporation
Systems Development Division, Poughkeepsie, New York

This is a tutorial paper[1] which highlights the influence of computers not only on the modus operandi of circuit design but also on network theory. The impact of computers on circuit design techniques is obvious enough from the proliferation of network analysis programs. But the influence of this new technology on network theory itself is perhaps more subtle.

In one direction, the computer is helping to popularize the use of matrix notation in network theory because this notation is valuable in programming and inherently elegant for describing network problems. In another direction, the computer is hastening the recognition of the broad scope of network theory as exemplified by programs employing a network approach to mechanical and structural analysis. Finally, the computer is forcing the development of analytical and numerical methods which are particularly well adapted to both the requirements and the capabilities of automatic digital computation.

This paper stresses the importance of keeping the peculiarities of the computer strongly in mind when choosing techniques and methods of analysis instead of blindly using the computer to implement traditional methods. Only in this way can the real power of the computer be properly matched to the problems which it is called upon to solve for the circuit designer.

In keeping with this philosophy, a brief review is given of a matrix-topological formulation of the network problem which has been used by such programs as NET-1, ECAP, and STRESS -- the latter having been developed for structural analysis. This formulation, which is applicable to networks of any size and complexity, is based on the pioneering work of Gabriel Kron and on the more recent contributions of Roth.

From a physical point of view, the network problem is concerned with predicting the behavior of a system of interconnected elements in terms of the element characteristics and the manner of interconnection of these elements. Mathematically, the network problem involves the properties of an underlying topological structure, or linear graph, and a superimposed algebraic structure consisting of the interrelations between the network variables. The matrix approach nicely correlates these two points of view and precisely identifies the separate roles of the element characteristics and of the interconnections in determining the overall behavior of the network.

The bookkeeping chores associated with the interconnection properties of the network, and expressed as Kirchhoff's laws, are completely and automatically handled by the use of four topological matrices: the branch-node connection matrix A, the node-to-datum path matrix B_T (for the tree of the network), the circuit matrix C, and the cutset matrix D. These matrices are intimately related to one another and can be derived in sequence by easily programmed algorithms from a table of branch node connections.

Ohm's law may also be expressed in matrix form wherin the off-diagonal matrix terms account for the so-called "dependent sources" that are frequently used to describe the behavior of active devices. Thus active networks are as readily handled as passive ones. The interrelations between the variables involved in expressing Ohm's law and Kirchhoff's laws comprise the algebraic structure of the network problem and are neatly summarized by a transformation diagram developed by Roth.

The analysis of the electrical network problem by the classical mesh, node, and cutset methods is reviewed next. This is followed by an explanation of the mixed method of analysis which underlies both Bashkow's A-matrix treatment of the transient problem and the so-called "state variable" approach. It is shown that the mixed method is applicable to d-c and a-c network problems as well as to transient problems. Thus, there are four general methods of analysis -- or formulation. Moreover, all of them can be programmed so as to derive the network equations automatically from simple input data describing the network parameters and configuration.

Numerical techniques for solving both linear and nonlinear d-c problems are discussed briefly. In particular, the work of Sato and Tinney is cited as an example of an advance in the field of electrical power system analysis which electronic engineers

have overlooked. These authors, being forced to handle problems an order of magnitude larger tban the largest usually encountered in electronics, have developed techniques for taking full advantage of the sparsity of the nodal admittance matrix. As a result, they can solve huge systems of equations (1000 or more variables) in short order and without exceeding the core storage capacity of a typical large scale computer.

The utility of Kron's method of interconnecting solutions is pointed out in connection with parameter studies and a simplified description of the method is given. Although Kron's method can also be used to derive sensitivity formulas, an easier approach based on the differentiation of the nodal equations is presented. Finally, the solution of nonlinear problems by the Newton-Raphson iteration method and by a method developed by Katzenelson is discussed.

A new approach to the solution of a-c problems is described next. Using the equations derived from the mixed method, the inverse matrix $(sU - A)^{-1}$ is computed in terms of the eigenvalues and eigenvectors of the (Bashkow) A matrix, s being the complex frequency parameter and U a unit matrix. Solution of the eigenproblem for the matrix A defines a transformation which diagonalizes not only A but also the matrix (sU - A). As a consequence, the problem of computing the inverse of (sU - A) is reduced to the trivial task of inverting the diagonal matrix $(sU - \Lambda)$, where Λ is the (diagonal) matrix of eigenvalues of A, followed by some matrix multiplications of simple form.

This approach leads to a partial fraction expansion for any of the state variables or network functions. It permits the straightforward computation of the sensitivities of these variables or network functions with respect to frequency or any network parameter. Finally, the sensitivities of the network poles (which are identical with the eigenvalues of A) can be computed.

Since this new method of a-c analysis can be extended to the treatment of linear transient problems and yields essentially the same theoretical information as the traditional Laplace transform approach, it is recommended as an alternative thereto, particularly for implementation on a computer. The Laplace transform technique suffers from two serious disadvantages when applied to computer-sized problems. First, the job of computing the coefficients of the polynomials in a network dunction P(s)/Q(s) is prone to serious inaccuracies, especially when the polynomials are of high degree, and may be very time-consuming. (For example, the "topological formula" technique for computing these coefficients may involve identifying and calculating admittance products of millions of trees for networks of even modest size.) Second, the roots of the polynomials themselves may be extremely sensitive to errors in the coefficients.

By contrast, the calculation of eigenvalues and eigenvectors, particularly by the Q-R algorithm of Francis, is economical (about 10 times the cost of inverting a matrix) and computationally stable, except possibly for matrices with a very wide spread of eigenvalues. Moreover, the Q-R method readily handles the problem of multiple eigenvalues. This is not to say that the eigenvalue approach is entirely free of difficulties. But it appears to be far better adapted to the computer and faster than the Laplace transform technique.

After a description of the analytic solution of the linear transient problem in terms of eigenvalues and eigenvectors, an explanation of the principal obstacle encountered in the numerical integration of differential equations is given. It is pointed out that the usual integration schemes, such as Runge-Kutta and the predictor-corrector methods, are limited by the fact that the integration step size must be kept comparable to the smallest time constant (which is the reciprocal of the largest eigenvalue) of the network in order to avoid numerical instability.

This limitation can be severe in networks having a large ratio of largest to smallest eigenvalue. For the largest eigenvalue (smallest time constant) controls the permissible integration step size whereas the smallest eigenvalues (largest time constants) are principally responsible for the behavior of the network and determine the interval over which the integration must be carried out. Although there are some implicit integration formulas which are free of this restriction, they are of low order accuracy and require the solution of a system of linear equations at each time step. Hence, they are considered unsatisfactory for general use.

A quasi-analytic approach to this problem is discussed in whicb the diagonal terms of the A matrix are handled analytically and the off-diagonal terms numerically. The stability of tbis method is governed by the largest eigenvalue of the off-diagonal part of the A matrix -- and this eigenvalue may be appreciably smaller than the largest eigenvalue of A. Even so, by means of a similarity transformation on A, tbe largest eigenvalue of the off-diagonal part of the transformed matrix may be made arbitrarily small. Thus, the integration step size can be increased significantly without producing

numerical instability.

The effectiveness of this technique is illustrated by a simple example but the problem remains to find an economical method of determining and applying the appropriate transformation(s), particularly in the case of nonlinear networks.

In any event, the route that will eventually permit the small time-constant barrier to be breached appears to lie in the direction of performing meaningful operations on the A matrix itself rather than accepting this matrix as given and trying to devise clever integration formulas. For this matrix clearly holds the key to the entire integration problem.

The paper concludes by pointing out some of the directions in which the next generation of network analysis programs should advance such as: incorporating improved analytical and numerical techniques, providing facilities for statistical analysis, storage and retrieval of device models, and the handling of much larger networks -- including transmission lines. It is also acknowledged that the advent of integrated circuits is already placing strong demands on existing analytical, numerical, and programming techniques and will inevitably force the development of more powerful methods for solving partial differential equations.

Finally, the desirability of evolving from the present stage of network analysis to the next stage of optimization is recognized, but it is concluded that the engineer himself will continue to play an essential role in the design process for some time to come. Although the computer will certainly replace the routine computational tasks of the engineer, it will enhance rather than supplant his creative abilities.

Reference

1. Presented at the IEEE International Convention, March 20-23, 1967, in New York City. Available from the author as IBM Technical Report 00.1562.

OLCA: AN ON-LINE CIRCUIT ANALYSIS SYSTEM

By

DR. HING C. SO

Dr. So was born in Canton, China. He received the B. S., M. S., and Ph. D. degrees in Electrical Engineering from the University of Illinois in 1956, 1957, and 1960, respectively.

From 1960 to 1965 he was on the faculty of the Department of Electrical Engineering at the University of Rochester. During the academic year 1963-64 he was on leave from the University of Rochester and was at the Bell Telephone Laboratories, Murray Hill, N. J.

In September 1965 he left an appointment of Associate Professor at Rochester University to join the Bell Telephone Laboratories at Murray Hill, where he has been a member of the technical staff in the Information Processing Research Department. His current interest lies in computer applications and network theory.

OLCA: AN ON-LINE CIRCUIT ANALYSIS SYSTEM

By Hing C. So

Member of Technical Staff
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

SUMMARY

A software system for on-line analysis of linear networks is described. Inputs to the system such as circuit diagrams are given via a light pen and a teletypewriter. Analysis results are available in various forms, including oscilloscope displays of frequency characteristics. The system guides the user through necessary steps by displayed comments; it also checks for user's errors.

Introduction

The application of the digital computer to network design is of widespread interest. However, current applications primarily use the computer in over-the-counter runs.[1] Thus input data are given on tapes or punched cards, usually according to rigidly specified formats. Furthermore, computation results from a circuit analysis program are available to the network designer only after an always-present delay, i.e., the turn-around time. To make the computer a genuinely effective tool in the iterative network design process, it is imperative that the coupling between the designer and the computer be as efficient as possible. This paper describes OLCA, a system of programs designed for on-line analysis of linear networks, which has existed at the Bell Telephone Laboratories since August 1966.

Typically, a user of OLCA draws the circuit under study with a light pen on the cathode ray tube of a special remote graphic console known as GRAPHIC 1. Numerical data such as circuit element values and frequencies are supplied through the teletypewriter of the console. If desired, trial circuits representing deviations from the initial circuit may be simultaneously specified. After the entire problem has been defined at the console, it is sent to the central computer via data links for analysis. At his option the user may obtain displays of frequency characteristics on the oscilloscope of the console and/or tables and plots through the conventional output facilities of the central computer. Thus a user at the console can maintain a continuous dialogue with the main computer, utilizing the central computing power rapidly and easily to implement his design.

A full description of the experimental console, GRAPHIC 1, has been given elsewhere.[2] The operating environment for OLCA is given in Fig. 1. The central computer used is an IBM 7094. The outstanding characteristic of GRAPHIC 1 is that it contains a small local computer, a PDP5, which is used to direct all problem-defining activities at the console. The central computer is called into service only when circuit analysis is performed. This computing system is in contrast to that used by the two on-line circuit analysis programs reported at MIT,[3,4] where the Project MAC central computer is used to serve all console activities. The obvious advantage of not having to interrupt the central computer for every local task has many ramifications, which OLCA was designed to exploit fully. As a consequence, OLCA has the following features:

(1) No control button board is used. Instead, words, called light buttons, are displayed on the oscilloscope, as shown in Fig. 2. Whenever a light button is pointed at by the light pen, programs written for the local computer enable the user to perform the corresponding function, for example, to delete unwanted material. Thus the user can focus his attention on the oscilloscope; operation proceeds smoothly, and many accidental user errors that otherwise may occur are avoided.

(2) At any stage of the problem-defining process, OLCA displays only those light buttons that the user needs. In addition, at any time if any action is expected of the user, a guiding comment is displayed. The user cannot proceed without heeding the instruction. The result is that the user is steered step by step through all operations. No complicated operating rules need to be memorized by the user.

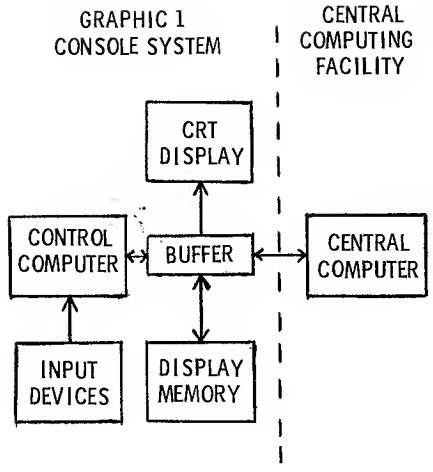


Figure 1.-The operating environment of OLCA

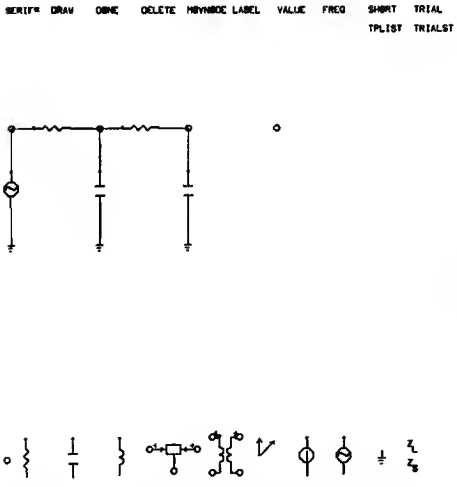


Figure 2.-A circuit being constructed. The words across the top of the oscilloscope are light buttons.

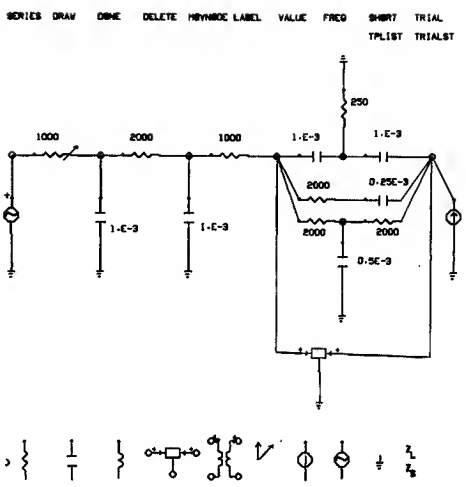


Figure 3.-A completed circuit. In this example the three-terminal device is an operational amplifier.

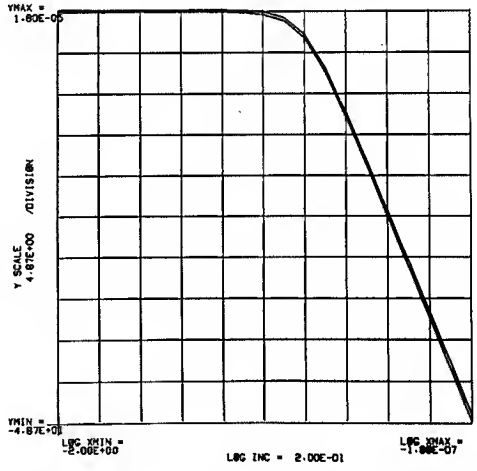


Figure 4.-The magnitude characteristics of the RC-active low-pass filter of Figure 3 and a deviation circuit, as seen on the oscilloscope.

(3) OLCA checks for most accidental user errors locally. For example, if at some stage the user is expected to select a node, the local computer checks the displayed item selected to prevent the accidental selection of a wire or a component by the light pen.

Because of these features in OLCA, a user with little knowledge of the system, and indeed with little experience in computing, can learn to use the system easily. The aim of providing efficient and easy interaction between the network designer and the central computer is accomplished.

It is evident from the preceding discussion that OLCA consists of three parts:

(a) A problem-defining program written for the PDP5 computer of GRAPHIC 1;

(b) A control program written for the IBM 7094 for data conversion, command execution and result display;

(c) A circuit analysis program[5] written for the IBM 7094 to perform frequency analysis of networks. A brief description of these programs is given. A discussion on the data structure is also presented.

References

Kuo, F. F.: Network Analysis by Digital Computer, Proceedings of the IEEE, vol. 54, No. 6, June 1966, pp. 820-829.

Ninke, W. H.: Graphic 1 - a Remote Graphical Display Console System, Proceedings of the Fall Joint Computer Conference, vol. 27, Spartan Books, Inc., Washington, D.C., 1965, pp. 834-846.

Katzenelson, J.: AEDNET: A Simulator for Nonlinear Networks, Proceedings IEEE, vol. 54, November 1966, pp. 1536-1552.

Dertouzos, M. L. and Therrien, C. W.: Circal: On-Line Analysis of Electronic Networks, Report ESL-R-248, Electronic Systems Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, October 1965.

So, H. C.: Analysis and Iterative Design of Networks Using On-Line Simulation, Chapter 2, System Analysis by Digital Computer, Edited by F. F. Kuo and J. F. Kaiser, Wiley, New York, 1966.

AUTOMATIC MASK AND WIRING PATTERN GENERATION

By

DR. HARLOW FREITAG

Dr. Freitag received his B. S. degree from New York University in 1955 and his M. S. and Ph. D. degrees from Yale, the latter in 1959. He joined IBM at that time and transferred to the Research Division in 1961.

He is interested in the use of computer-aided electronic design, particularly in the exploitation of integrated circuit technology.

He is currently manager of the Design Automation group.

DR. WILM E. DONATH

Dr. Donath was born in Mannheim, Germany. He was awarded his B. S. degree in Chemical Engineering from Carnegie Institute of Technology in 1954 and his Ph. D. degree in Chemistry in 1958. He did post-doctoral study at Polytechnic Institute of Brooklyn from 1958 to 1959.

From 1959 to 1961, he worked on atomic computations, and from 1961 to 1963, on molecular computations, both while at the IBM Watson Research Center, New York City.

Since 1963, Dr. Donath has been working on aspects of design automation problems.

AUTOMATIC MASK AND WIRING PATTERN GENERATION

H. Freitag and W. E. Donath
IBM Watson Research Center
Yorktown Heights, New York

SUMMARY

The design and fabrication of mask sets for integrated circuit manufacture has become a critical problem. A set of techniques utilizing a computer for the producing of such masks is described. One path is provided for the masks required to fabricate the devices and circuits. Other techniques have been developed to design and fabricate the wiring pattern used to interconnect the circuits into a functional array. One such technique employes the Programmed Interconnect Process (discretionary wiring) and is used when high levels of integration are desired for chips having relatively low individual circuit yields.

By the use of these techniques it is possible to design a system which will accept circuit schematics as input and automatically produce all the masks needed to fabricate integrated circuit chips in order to implement the given circuit structure in hardware. Furthermore, it is possible to incorporate this automatic scheme into a larger system which would include such facilities as circuit analysis and simulation.

INTRODUCTION

The exploitation of integrated circuit technology to implement electronic hardware faces many challenges. One of the most serious obstacles is the artwork needed to generate masks required for the fabrication of these complex semiconductor structures. It is certain that with integrated circuit technology, we will have to design and fabricate an enormous number of part types. It is also certain that chips will be produced with higher and higher levels of integration. The masks will thus be more complex and the likelihood of error in the design or fabrication of the mask sets is very great if manual techniques alone are employed. Moreover, there has been a growing need to decrease the turnaround time required from the initial design request for a new part until its actual fabrication.

While it has been customary for the artwork to be designed by specialists who may do little else, there is a growing trend today in the need for such designs to be generated by inexperienced personnel. Finally, there has been a growing sophistication in circuit analysis, design and fabrication by automatic techniques. Manual mask design and fabrication represent obstacles to a unified system for the design, analysis, and fabrication of digital and analog componentry by completely automatic techniques.

Each of these considerations argues for a computer-assisted process for artwork design and fabrication. Taken together, the need is overwhelming. Two sets of techniques were developed. One for designing and fabricating the artwork and the other to design and fabricate the interconnections between the circuits. Let us consider each of these techniques in turn.

DEVICE AND CIRCUIT MASKS

The automatic artwork generating technique contains three parts: A language to permit the designer to describe the patterns and structures required in the individual masks in the mask set to be used to fabricate devices and circuits; a computer procedure for translating the language into commands and finally, the automatic equipment which takes the commands and generates the masks themselves.

The language itself is a convenient shorthand and symbolic way of describing the patterns a designer wishes to generate with a minimum likelihood of error. The language is easily learned but extremely powerful. Inexperienced personnel, that is, individuals who are not familiar either with designing masks or programming have learned the language within two days and have produced successful masks within another two days.

The central feature of the language is a facility for defining individual structures and groups of structures on the mask and assigning a unique symbolic name to each group. The system is very analogous to that found in programming where a programmer defines his variables with symbolic mnemonic names. The programmer can then manipulate these names and leave to the computer the task of relating the names to the actual items of data. The language permits any structure or groups of structures to be defined by means of sub-structures. One defines a structure which will generate the base of a transistor. This structure is then used as part of the set of structures defining the transistor itself. The transistor structures in turn can be used to define a still

larger structure, say, a logical decision gate. The transistor definition can be used as parts of the definitions of other structures which require such a type of transistor. This hierarchy of definitions may be continued indefinitely. Such a facility permits the gradual accumulation of a library of parts which can be used over and over again in different mask structures. Once a given structure has been defined it need not ever be redefined. Thus, in time, masks can be designed by appropriately choosing pieces of masks which have been previously designed and have been shown to work successfully in another and perhaps unrelated application.

In general, any device which one wishes to construct in integrated circuit technology will require a number of masks. Another feature of the language permits any structure or group of structures to bs represented "three-dimensionally." When a designer requests a given pattern to be located at a particular part of a chip layout, the language automatically accounts for the multiplicity of masks required. The method by which this feature is implemented will be discussed below.

The language has a replication feature which permits a given pattern to be repeated at a fixed interval around the layout. The replication can be either a row or column vector or a matrix array. Since any pattern or group of patterns is defined relative to an arbitrary origin, the replication is effected by adding multiples of the specified increment to the coordinate specified in the pattern definition.

To initiate the process of designing the masks, an accurate representation of the layout is first drawn. With a little practice on the designer's part, the drawing need not be highly accurate but may be little more than a rough sketch. The designer then writes down the set of statements needed to produce the layout. This set may consist largely of previously prepared statements from other mask designs. The set of statements is punched on cards and fed to a computer which produces a digitally plotted representation of the masks which would be generated by the input statements. If any errors are found in the digital plot, the statements are corrected and a new plot of the mask is obtained. This correction procedure is repeated until the design is error-free. At this stage, the designer may call on the computer to produce a magnetic tape containing the commands necessary to generate the mask set. As mentioned before, each statement or statement group may refer to several masks simultaneously. One task of the computer is to remember on which mask layer the pattern is to be found and then sort the entire set of patterns mask by mask. Individual masks will then be produced with tbe appropriate patterns on them.

The actual generation of the mask is performed by an automatic exposure device. In our case we have used a device called a light table. The apparatus is similar to an ordinary microscope. (Figure 1) A high resolution photographic plate is placed on what corresponds to the stage of the microscope. A light spot is imaged on the plate by projecting it down the barrel of the microscope. The spot is turned on and off by an electronically controlled shutter. The mechanical stage is capable of motion in the X and Y directions and is controlled by a pair of digital stepping motors. The commands generated by ths computer are decoded and fed to the motors and the shutter to draw the required pattern on the photographic plate. In our experiment, the masks are drawn ten times size, the resulting plate is then used in a step-and-repeat camera which reduces the image a factor of ten and repeats the mask pattern so that a semiconductor wafer containing identical chips may be produced.

There are clearly many advantages to this system. The designer can work in what is for him a more natural environment while the computer assumes the responsibility for the tedious bookkeeping and numerical details where most of the errors are made. The ability to gradually build a library containing previous designs reduces the need for re-inventing devices and circuits while considerably speeding up the design process. The replication feature requires the designer to furnish a minimum of information, further reducing the possibility of error.

WIRING PATTERN GENERATION

The method of generating the mask required to produce a metallized interconnection pattern to connect the circuits into functional arrays may be handled somewhat differently than in the case of device and circuit masks. One reason is the greater capability for completely automating the design of these interconnection masks. Furthermore, if we wish to obtain the highest possible level of integration on the chip, a technique for generating the interconnection pattern in the presence of faulty circuit areas is required. The process that was developed to cope with the latter circumstance has been called the Programmed Interconnection Process (PIP).

It was recognized that the current yields of semiconductor devices and circuits do not per-

Figure 1. - Experimental Light Table

mit the interconnection of very many circuits directly on the chip to produce high levels of integration. In the PIP process, sometimes called discretionary wiring, a semiconductor wafer is fabricated with an array of circuits but no wiring personality. Each individual circuit is tested by a probe technique and the results of the test are fed to a computer which then determines which circuits are satisfactory and which are not. The computer then designs an interconnection pattern which wires together only the usable circuits. The output of the computer is a magnetic tape which controls a light table as before. In this case, however, the semiconductor wafer, having been coated with a metallization layer and then a photoresist layer, is placed on the stage of the light table. The photoresist is exposed directly, hence no mask is needed to produce the interconnection pattern. In this process every wafer will have its own interconnection pattern determined by its particular pattern of good and bad circuit areas. The technique assures us that only functioning circuits will be interconnected to produce a functioning array.

In order for the PIP process to be competitive on a cost basis, the costs unique to the PIP process, that is, testing the circuits, using the computer to generate an interconnection pattern and actually fabricating that pattern must be compatible with the other manufacturing costs. A major fraction of these unique costs is, of course, the computer time. Our PIP program runs on an IBM 7094 and provides interconnection patterns at a rate of about 0.5 sec per circuit interconnected. Furthermore, the program must be flexible enough to provide an interconnection pattern for every wafer, or almost every wafer, independent of its particular pattern of good and bad circuit areas. Since the PIP program must also be very fast, it is likely to be rather inefficient in its use of available interconnection area. The requirement that almost every wafer be wirable further increases the amount of interconnection area which is needed on the wafer. Hence, it is likely that a wafer layout intended for PIP will have a high ratio of interconnection area to circuit area. If we are not interested in very high levels of integration, or if our circuit yield is reasonably high, then one can assume all of the circuit areas in an array are good and generate an interconnection pattern which is the same for every wafer of a particular part type. One can then simply discard chips which are inoperative due to non-functioning circuits. This has been called the Fixed Interconnection Process (FIP). Here one may have the computer spend a relatively long time for each part type to produce a highly optimized wiring layout. The commands for generating the results of a wiring pattern can be run on the light table to produce a mask in a similar manner to the circuit and device masks except that the layout language need not be used for the interconnection mask as long as a program exists for generating the wiring pattern automatically.

THE COMPLETELY AUTOMATED SYSTEM

By combining either the PIP or FIP wiring approach with the library facility of the layout language, a completely automatic system for the generation of artwork for all the masks required is obtained. The input to this system would only be the circuit schematics to be implemented. To visualize the process we refer to Figure 2. A program can be written to assign circuit types

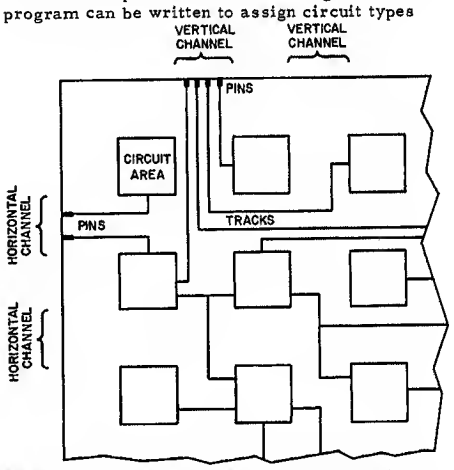


Figure 2. - Schematic view of an integrated circuit wafer. Circuit areas are fabricated in a rectangular array while the interconnections are in vertical and horizontal channels.

to the circuit areas shown in the figure. Since we are given the circuit structure to be implemented, this assignment can be performed to make most efficient use of the layout. We can, for example, minimize interconnection distance between circuits or improve the wirability of the layout by avoiding wiring congestion. Knowing what circuit type is to be located in each of the circuit areas, we can use the layout language to call out a previously designed group of patterns producing that circuit type and locate the pattern in the given circuit area. This is enough information to allow us to construct the masks which will fabricate that particular wafer. The interconnection pattern may be generated using either the FIP or PIP

program discussed above. The entire process from circuit schematic input to finished masks can take place entirely witbin the computer without any manual intervention being required.

REFERENCES

1. Freitag, H.: Design Automation for Large Scale Integration, WESCON, Los Angeles, Calif., August 25, 1966.

2. Triebwasser, S.: Programmable Interconnection Techniques, ISSCC, University of Pennsylvania, Feb. 11, 1966.

3. Lathrop, J. W.: Discretionary Wiring Approach to Large Scale Integration, WESCON, Los Angeles, Calif., August 25, 1966.

4. Freitag, H.: Automatic Mask Generation for Integrated Circuits, NEREM Conference, Boston, Mass., Nov. 4, 1966.

5. Critchlow, D.: Layout and Mask Generation in Large Scale Integration, IEEE International Convention, New York, N. Y., Mar. 23, 1967.

THE DESIGN OF DIGITAL FILTERS

By

DR. JAMES F. KAISER

Dr. Kaiser received his E. E. degree from the University of Cincinnati in 1952; and his S. M. and Sc. D. degrees from the Massachusetts Institute of Technology in 1954 and 1959, respectively. He was a member of the faculty of the Massachusetts Institute of Technology from 1956-1960 and a member of the Bell Telephone Laboratories staff from 1959 to present.

He has been concerned with problems of data processing, digital filter design, system simulation, and computer graphics.

He is co-author of two books--"Analytical Design of Linear Feedback Controls" with G. C. Newton and L. A. Gould, and "System Analysis by Digital Computer," with F. Kuo.

Dr. Kaiser is a member of IEEE, the Association for Computing Machinery, the Society for Industrial and Applied Mathematics, and Eta Kappa Nu, Sigma Xi, and Tau Beta Pi.

THE DESIGN OF DIGITAL FILTERS

By James F. Kaiser

Member of Technical Staff
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

SUMMARY

The simulation of dynamic systems and continuous filter networks and the filtering or processing of data signals by means of digital computers require both the design and utilization of digital filters.* In simulation work the digital filters are designed to represent or approximate continuous systems such that the behavior of the continuous systems can be evaluated, either numerically or subjectively, for a wide range of input signals. This is in contrast to the more usual network transient response calculations based on very simple inputs such as steps, ramps, impulses, and single sinusoids.

The effective use of the discrete models depends on a detailed knowledge of the nature of the discretizing approximation process used, of the sampling-frequency vs. simulation-accuracy tradeoff, and of the essential linearity or nonlinearity of the continuous systems being modeled. This paper is, therefore, concerned with a brief discussion of two of the more commonly used digital filter design methods. This is followed by a general description of a digital filter design program which is capable of generating discrete representations for linear continuous filters including a wide variety of the classic filter forms. The paper concludes with comments on the application of the program.

Introduction

Digitalizations may be sought for both linear and nonlinear systems. For nonlinear systems the choice of sampling frequency may have to be made quite large to yield sufficient accuracy in the simulation. Here the amount of calculation required may easily approach that involved in directly integrating the differential equations by standard numerical techniques. On the other hand, the combination of the superposition property of linear systems and the quasi-bandlimited nature of realistic input signals makes possible the efficient simulation of systems at sampling rates approaching the Nyquist sampling limit.

Linear continuous systems which are describable by linear differential equations with constant coefficients become representable by linear difference equations when in the discrete form. Thus no inherent additional complexity is added by going from the continuous domain to the discrete domain of representation [1]. Numerous techniques of digitalization have been described in the literature [2]. Of all these techniques, the two most widely used in the simulation of continuous filters and networks are the standard z transform and the bilinear z transform. These two methods differ in the nature of the approximation made in going from the continuous domain to the discrete domain.

Design Methods

Viewed from the frequency domain, the approximation made in using the standard z transform is easily seen from the defining equations

$$H^*(s) = \sum_{n=-\infty}^{\infty} H(s+jn\omega_s) + \frac{T}{2} h(0^+) \qquad (1)$$

and

$$H^*(z^{-1}) = T \sum_{n=0}^{\infty} h(nT) z^{-n} \qquad (2)$$

*The term digital filter refers to the computational process or algorithm by which a sampled signal or sequence of numbers acting as an input is transformed into a second sequence of numbers termed the output signal.

Thus the impulse response of the obtained digital filter, $H^*(z^{-1})$, is identical to the samples, $h(nT)$, of the impulse response of the continuous filter $H(s)$. However, the frequency response characteristic, $H^*(s)$, of the digital filter in general differs from that of the continuous filter, $H(s)$, by the addition of the terms $H(s+jn\omega_s)$, $n \neq 0$, this effect being termed folding or aliasing. Unfortunately, when $H(s)$ is a rational function of s, it is not band-limited and therefore $H(s) \neq H^*(s)$ in the baseband $\left(-\frac{\omega_s}{2} < \omega < \frac{\omega_s}{2}\right)$. This is especially true for many filter designs such as elliptic and Chebyshev Type II filters for which the high frequency behavior is either a constant or at most K/s. Application of the standard z transform technique consists simply of first obtaining a partial fraction expansion of $H(s)$ in its poles and then z transforming each of the individual terms using transform pairs derived from

$$\frac{1}{s+a} \rightarrow \frac{T}{1-e^{-aT}z^{-1}} \tag{3}$$

where $z^{-1} = e^{-sT}$.

A digitalizing technique which does not introduce error due to aliasing is the bilinear z transformation defined by

$$s = \frac{2}{T} \tanh \frac{s_1 T}{2} \tag{4}$$

which becomes upon substituting $z^{-1} = e^{-s_1 T}$

$$s = \frac{2}{T} \frac{(1-z^{-1})}{(1+z^{-1})} \tag{5}$$

Thus, in terms of the z^{-1} plane, this algebraic transformation has the property of uniquely mapping the left-half of the s-plane into the exterior of the unit circle in the z^{-1} plane. Folding errors are eliminated since no folding occurs.

The bilinear z-form is applied simply by making the substitution indicated by (5) in the transfer characteristic $H(s)$. Hence,

$$H^*(z^{-1}) = H(s)\Big|_{s = \frac{2}{T}\frac{(1-z^{-1})}{(1+z^{-1})}} \tag{6}$$

The price paid for this simplicity in digitalization is the nonlinear warping imparted to the frequency scale as can be seen by setting $s_1 = j\omega_1$ and $s = j\omega$ in (4) to yield

$$\frac{\omega T}{2} = \tan\left(\frac{\omega_1 T}{2}\right) \tag{7}$$

Because of this frequency warping aspect (especially for frequencies greater than $2/T$), the bilinear z-form is most useful in obtaining digital filter approximations for continuous filters whose magnitude characteristics are essentially piecewise constant in successive pass and stop bands. This type of frequency behavior is typical of many lowpass, bandpass, and bandstop filters. Compensation can be made for the effect of warping by prewarping the continuous filter design in the opposite way so that upon applying the bilinear z form, the critical frequencies will be shifted back to the desired values [3].

A Filter Design Program

The versatility of these two design procedures has been utilized effectively by Golden and Kaiser in the implementation of a general digital filter design program. The program has been further refined by Golden to couple its results effectively to a simulation program for complex systems. The basic scheme of the design program is shown in Fig. 1.

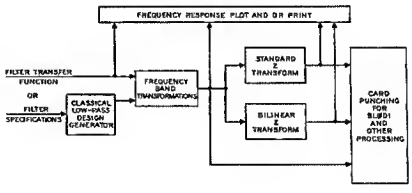


FIG.1

The program is divided into essentially two parts: the first part generates the linear continuous filter design required to meet the filter specifications, while the second part comprises the digitalization procedure which yields the desired coefficients for the resulting digital filter design. For input the user may specify either 1) the actual continuous filter transfer characteristic whose digitalization is desired or 2) simply the general type of filter characteristic desired and the specifications associated with that filter type. The program can generate the basic lowpass filter types - Butterworth, Bessel, transitional (Butterworth-Thomson), Chebyshev Type I and Type II, and the elliptic or Cauer parameter types - from specifications such as in-band ripple, out-of-band loss, transition ratio, filter type, band-edge frequencies, and order, if applicable. The program then has facility for transforming any of these lowpass designs to bandpass, bandstop, or highpass filter designs by means of the standard frequency band transformation. The basic lowpass designs and band transformations are discussed in detail in many network synthesis references, for example Weinberg [4]. Provision is also included to prewarp automatically the original filter band-edge frequencies and thus give a prewarped continuous filter design if the bilinear z transform is to be used.

The digitalization section of the program gives the user the choice between the standard z transformation and the bilinear z transform with or without warping correction. The results of the digital filter design are printed out as the coefficients of the digital filter in the parallel canonical form

$$H^*(z^{-1}) = A_{00} + \sum_{i=1}^{m} \frac{A_{1i}z^{-1} + A_{0i}}{B_{2i}z^{-2} + B_{1i}z^{-1} + 1} \qquad (8)$$

which has been shown to be a preferred realization form from accuracy considerations [5], [2, see especially pp. 271 ff]. Also available to the user are both tabulated listings and complete graphical plots of the magnitude- and phase-versus-frequency characteristics of not only the continuous filter design but also of the final digital filter design. This allows a critical comparison between the characteristics of the obtained digital filter and the continuous filter from which it was derived.

This program has been a valuable aid in determining the digital filter coefficients required for the simulation of complex systems via the block diagram compiler BLODIB [6]. When special purpose digital filters are to be physically implemented by hardware, the program described can be used to obtain a first set of values for the filter coefficients. From this set of values the designer can vary the bit size representation of the coefficients until he has obtained a satisfactory compromise between filter complexity and performance characteristics. The requirements on filter coefficient accuracy have been investigated at some length by Kasier [5]. Other investigators have started to look into the problem associated with roundoff and truncation in the carrying out of the arithmetic operations of the digital filter. The overall picture of error analysis is by no means complete as it depends also on the nature of the input signal and the performance criterion as well as the realization scheme.

In this paper the use of digital filters for data processing and simulation of complex systems has been motivated. Two digitalization procedures were briefly discussed. A digital filter design program was then described which incorporated the digitalization procedures. The paper concluded with brief comments on the range of application of the program.

References

1. Steiglitz, K.: The Equivalence of Digital and Analog Signal Processing. Information and Control, vol. 8, No. 5, October 1965, pp. 455-467.

2. Kaiser, J. F.: System Analysis by Digital Computer, New York, John Wiley and Sons, Inc., 1966, edited by F. F. Kuo and J. F. Kaiser. pp. 218-285.

3. Golden, R. M. and J. F. Kaiser: Design of Wideband Sampled Data Filters. Bell System Technical Journal, vol. 43, Part 2, July 1964, pp. 1533-1546.

4. Weinberg, L.: Network Analysis and Synthesis, New York, McGraw-Hill, 1962.

5. Knowles, J. B. and R. Edwards: Effect of a finite-word-length computer in a sampled-data feedback system. Proc. IEE, vol. 112, No. 6, June 1965, pp. 1197-1207.

6. Karafin, B. J.: A Sampled-Data System Simulation Language. Chapter 8 of System Analysis by Digital Computer, New York, John Wiley and Sons, Inc., 1966, pp. 286-314.

7. Rader, C. M. and B. Gold: Digital Filter Design Techniques in the Frequency Domain. Proceedings IEEE, vol. 55, No. 2, February 1967, pp. 149-171. See especially pp. 165-169.

NET-2 CIRCUIT ANALYSIS PROGRAM

By

ALLAN F. MALMBERG

Mr. Malmberg received his B.A. degree from St. Olaf College, Northfield, Minnesota, in 1953 and completed two years of graduate study in Physics at the University of Minnesota, Minneapolis. In 1955 he joined the staff of the Los Alamos Scientific Laboratory, Los Alamos, New Mexico. At Los Alamos he worked initially on problems of chemical instrumentation, and then developed a magnetic tape system and a high-speed coincident current magnetic memory for the MANIAC II computer.

He is the author of the NET-1 Network Analysis Program, completed in 1962, and the NET-2 program currently under development. He is a member of the IEEE and Sigma Pi Sigma.

NET-2 CIRCUIT ANALYSIS PROGRAM

by Allan F. Malmberg

Los Alamos Scientific Laboratory*
Los Alamos, New Mexico

SUMMARY

The general features of the NET-2 program are presented. This program, under development on the MANIAC computer, will perform nonlinear DC and transient analysis and linearized AC analysis of arbitrary circuit configurations.

The language for describing the circuit schematic is presented in detail. Features for inclusion of models of various linear and nonlinear devices are discussed. The program is written so as to permit device models to be added without rewriting NET-2 itself. Models and their parameter values are stored in a model library and called into the calculation automatically. Input facilities are included for superseding library values of device parameters.

A major featura of NET-2 is its variational capability. Three types of variational procedures are available: the STATE calculation which varies circuit parameters in discrete steps, the Monte Carlo calculation which determines the statistical distribution in response variables for a given circuit parameter population distribution, and the Extrema Search which extremizes the value of a response variabla within an allowable circuit parameter space.

INTRODUCTION

The NET-2 Circuit Analysis Program is currently under development on the MANIAC computer at the Los Alamos Scientific Laboratory. It is a logical extension of the highly successful NET-1 program which was completed in 1962. NET-1 performs nonlinear analysis of circuits whose topology is fixed using nominal values of the circuit elements; NET-2 extends the analysis to variational studies in the circuit element values as well as permitting changes in topology.

NET-2 observes the same philosophy of ease of usage as NET-1: the user describes his circuit in a simple engineering-oriented language, and he is not required to have any knowledge of mathematical techniques, programming languages, or computational methods in order to use the program successfully.

Many of the restrictions in NET-1 have been removed to permit a more natural description of the circuit. A circuit described to NET-2 may exist in several electrically isolated parts. Voltage and current sources which are floating in the circuit are handled in a straightforward manner. Furthermore, circuits are not required to contain either sources or impedances, thus permitting the handling of switching and path problems.

*This work performed under the auspices of the U.S. Atomic Energy Commission.

Circuit elements available in NET-2 include resistors, capacitors, inductors, mutual inductive couplings, switches, voltage and current sources, and a library of device models. The circuit elements may be interconnected without restriction, provided a physically impossible situation does not result (e.g., a zero impedance mesh of voltage sources).

CIRCUIT DESCRIPTION

The circuit to be analyzed is described by a sequence of entries, each of which is a description of a single circuit element. These entries may be written following a process of assigning, on the schematic diagram, identification symbols to the elements and names to the connection points.

Circuit elements are identified by an ID consisting of a letter prefix, designating the kind of element, and a non-zero integer. The integers need not be sequential and the same integer may be used in ID for different kinds of elements. The ID may not contain more than six characters.

Voltage and current sources are assigned ID in the form V16 and I4. Linear elements such as rasistors, capacitors, inductors, mutual inductive couplings, and switches are assigned ID in the form R34, C16, L34, K5, and S2. Modelled devices, such as transistors and magnetic cores, are assigned ID in the form X12, X7, etc., where X is the appropriate letter prefix for the particular devics.

All connection points are assigned arbitrary alphanumeric names consisting of a maximum of six characters each. The ground node is assigned the integer 0 as its name. It is not necessary that there be a circuit path between various isolated parts of a circuit.

If a source element is connected between a node and ground, it is convenient to give the node the same name as the source, such as V12, I6, etc. Such a designation allows a simplification in the description of the source element itself.

Parallel circuit segments exist whenever a circuit contains several identical segments, and these segments have all connection points into the rest of the circuit in common. NET-2 circuit element description formats have special provisions for describing such parallel segments. In describing the parallel segments, node names and IDs are assigned within only one of the segments. The element descriptions for that segment are then tagged with a number in parentheses to indicate how many parallel segments are represented.

The circuit element formats for resistors,

capacitors, and inductors are:

```
Rn  (p)  f  g  xxx
Cn  (p)  f  g  xxx
Ln  (p)  f  g  xxx
```

where Rn, Cn, Ln = element ID
p = number of parallel units (optional)
f and g = connection points
xxx = value of element in kilohms, picofarads, or microhenries.

The circuit element format for mutual inductive coupling is:

```
Kn  i  j  xxx
```

where Kn = coefficient of coupling ID
i and j = IDs of the inductors involved
xxx = value of the coefficient of coupling.

The switch is an ideal, zero impedance, single pole single throw device. It may be initially open or closed during a steady state analysis as well as operated at prescribed times during the transient analysis. The format is:

```
Sn  f  g  Mode
    Mode  xxx
    Mode  yyy
    Mode  zzz
```

where Sn = switch ID
f and g = connection points
Mode = status of the switch, specified by the words "Open" and "Close." The first mode designation, following the g connection point, is the setting of the switch during the steady state analysis and at the initial time of the transient analysis. Subsequent mode designations indicate opening and closure of the switch at the respective times xxx, yyy, zzz, etc., during the transient analysis.

Source elements may be constant in value or time varying. The general format for voltage sources is:

```
Vn  f  g  Value
```

Current sources may also include parallel segment information:

```
In  (p)  f  g  Value
```

where Vn and In = element ID
p = number of parallel units (optional)
f and g = connection points
Value = information regarding waveshape and associated amplitude and time parameters.

The value of the source is always referred to connection point g. If the name of connection point f is the same as the source ID, and connection point g is node 0, it is permissible to omit the specification of f and g in the source element description.

The Value portion of the source element description specifies the time behavior of the source. Sources may be constant in value, or be time dependent. Time dependent sources include repetitive trapezoidal pulses, nonrepetitive tabulated waveforms, decaying exponentials, and sine waves. In addition, a small signal sine wave is available for excitation of the AC steady state analysis. Both amplitude and phase of this AC sine wave may be specified.

NET-2 has the capability of utilizing modelled devices which may be defined at any time and placed in a model library. This process is a relatively simple one and does not necessitate the rewriting of the NET-2 program. The models may represent multi-terminal linear or nonlinear devices, such as voltage controlled current sources, silicon controlled switches, magnetic circuits, photosensitive devices, and entire functional circuit blocks.

Each modelled device is represented in the analysis by an equivalent circuit, a set of controlling equations for the equivalent circuit, and a set of device parameters for the coefficients of the controlling equations. The general format for device entries consists of the device ID, the number of parallel units, the connection points (as many as needed), a type name, and optional mode information. The ID prefix specifies the kind of device (e.g., klystron, nonlinear inductance, etc.) from which NET supplies an equivalent circuit and controlling equations. The type name enables NET to obtain particular parameter values for the equations so that different devices of the same family (e.g., 2N705, 2N706A, etc., in the case of transistors) may be represented in the calculation. The mode information specifies constraints which may be imposed on the device during analysis, e.g., the specification of the magnetic state of a magnetic core.

The device parameter values may be changed as part of the input by means of the PARAMETER entry. In this entry specific devices are indicated, followed by the parameter name and its new value, for example:

```
PARAMETER
    T35
            Bn  25
            Ics .001
    D3
            Temp 35
```

The circuit description provides for a nominal nonlinear DC steady state solution and a transient solution using the nominal DC solution for initial values. If the circuit contains an AC sine source and a specified frequency value for this source, a linearized AC steady state solution will also be produced. This linearized AC solution is obtained by first solving the DC nonlinear problem, evaluating the impedances at the DC

operating point, and then solving the small signal AC steady state problem using these impedance values.

VARIATIONAL CAPABILITIES

It is possible to generate additional AC and DC steady state solutions by varying source values, circuit element values, device parameter values, and circuit topology. There are several ways of accomplishing the variation, depending upon the nature of the study undertaken.

The STATE entry provides one method of modifying circuit values and topology. In this entry specific values for circuit elements and parameters are indicated. These new values supersede the nominal values. Any values not specifically entered in a STATE entry are taken from the nominal circuit description.

It is also possible to vary circuit parameters and frequency in a set of discrete linear or logarithmic steps in a STATE entry so as to perform a parametric study. It is possible to refer to the entire ensemble of any element kind by using only the letter prefix for that element. Changes in device parameters may be made with the same general format used in the PARAMETER entry. The order of listing all items is important since the last value listed is the one used.

Device modes may be specified, but such information must be listed separately from any parameter changes. The word NONE may be used to remove a previously specified mode.

An example of a STATE entry is:

```
STATE 5
   R7    1.5
   T   D2
   HG
         Temp 35
   T
         Bn   68
   T2
         Bn   45
   R  .1  (4)  .5
   R5  2.7
   T  OFF
   T2  NONE
   S12  CLOSE
```

This list is interpreted in the following manner:

Temp = 35 for diode D2, all transistors and all Hall generators
Bn = 68 for all transistors except T2
Bn = 45 for T2
The value of all resistors except R5 varies in 4 linear steps from .1 to .5, giving a total of 5 solutions. All resistors which are varying will have the same value during any given solution. Note that R7 is not held at 1.5 because the variational entry for all resistors supersedes the R7 entry.
R5 = 2.7
All transistors except T2 are held off. Note that the NONE tag cannot be combined with the substitution for Bn of T2.
S12 is closed.

It is possible to generate two dimensional studies (or higher dimension studies) as in the example:

```
STATE 3
   ØV12    -30 (4) -60
   T45
         Bn  75
         Ics .001 (5)  .002
```

In this case the phase of source V12 and the Ics parameter of T45 will be varied over a two dimensional grid of size 5x6, giving a total of 30 solutions.

If a logarithmic variation is desired the number of steps is suffixed with an asterisk:

```
STATE 57
   FREQ  .001 (25*) .1
```

In this example the frequency of the AC sine sources is varied logarithmically from .001 to .1 in 25 steps.

Another variational calculation available in NET-2 is the Monte Carlo calculation. Here NET randomly picks circuit parameter values from known population distributions, constructs and analyzes the resultant circuit, and then tabulates the population distribution for the circuit responses of interest. The resultant circuits will then have response distributions which are statistically identical to the actual circuit in mass production.

In order to accomplish the Monte Carlo calculation, NET must know the population distribution shapes for the circuit parameters, the numerical values at the end points of the distributions, and the number of circuits which should be analyzed to give the representative statistical behavior.

Distributions available include flat, triangular, Gaussian, and tabulated distributions. Distributions may be specified in either linear or logarithmic space. Limit values may be specified as either actual values or relative (percentage) values. Population distributions for device parameters may be stored as part of the model library.

A third variational capability in NET-2 is the Extrema Search. In this type of calculation NET attempts to extremize the value of a particular response variable by varying circuit parameters within an allowable parameter space. Since the extrema point for a given response variable may not be at the limit values of the circuit parameters, this method is more powerful than the so-called worst case analysis. It is also valuable in the optimization of circuit response. Both DC and AC responses may be treated with this calcula-

tion.

TRANSIENT ANALYSIS

Transient analysis always proceeds from the initial values calculated by the nominal DC analysis. The transient analysis can be terminated automatically when a termination condition has been satisfied. The termination condition may be composed of simple Boolean functions of a variety of criteria, including the switching status of devices, values of response variables, and elapsed real time.

INPUT-OUTPUT

Input-output capabilities depend upon hardware availability in specific computers. The pilot version on the MANIAC computer has input via cards, paper tape, and on-line keyboard. Output may be in printed or graphical form. Transient solutions may be displayed as computation is in progress. Graphical displays include linear, semilog, log-log, and histogram plots for all response variables, node-to-node impedances, sensitivity coefficients, and functions of these quantities. Variational steady state analyses may be displayed in parametric form.

AUTOMATIC TRANSIENT/A-C SENSITIVITY
ANALYSIS WITH APPLICATIONS

By

DR. J. VENN LEEDS, JR.

Dr. Leeds attended Rice Institute from 1951 through 1956 and received both B. A. and B. S. E. E. degrees. He attended the University of Pittsburgh from 1957-1963, and received E. E. and Ph. D. degrees.

Dr. Leeds is presently Assistant Professor of Electrical Engineering and Environmental Engineering, Rice University. He began his career as an Engineer with Bettis Atomic Power Division, Westinghouse (June 1956 - May 1963). Later he became a Consultant with Esso Production Research Company, (May 1963 to present). He is also a Consultant with Gulf Aerospace Corporation.

Dr. Leeds holds memberships with the Institute of Electrical and Electronic Engineers (IEEE); and IEEE Professional Groups on Circuit Theory; Information Theory; and Automatic Controls.

AUTOMATIC TRANSIENT/A-C SENSITIVITY ANALYSIS WITH APPLICATIONS

By J. V. Leeds, Jr.

Assistant Professor, Electrical Engineering
Rice University
Houston, Texas

SUMMARY

A new method of sensitivity calculations for single parameter and multiple parameter sensitivities has been developed. The method has a direct interpretation in terms of network configuration. Furthermore, it may be used on existing digital computer network analysis programs without modification. The ease with which multiple parameter sensitivity calculations can be made has been used to analyze continuously equivalent networks and the mathematical model of the human cochlea.

INTRODUCTION

In performing a sensitivity analysis, it is often desired to calculate the sensitivity function of a large number of response variables as a function of a single parameter. Conversely, the sensitivity function of a single response variable to changes in a large number of parameters may be desired. Efficient solutions to both problems are presented in this note. The solution to the first problem is quite general in applicability. The solution to the second problem is limited to AC calculations for reciprocal networks unless one wishes to do convolutions.

One distinct advantage of the method is that currently available digital network analysis programs, such as ECAP and NET-1, can be used without modification for sensitivity calculations.

Although examples in this paper are limited to electrical networks, any analagous systems may be analyzed by the same method. The sensitivity functions in this paper are obtained with respect to the fundamental parameters of the network, resistance, capacitance, inductance, and current gains.

DEVELOPMENT OF SINGLE PARAMETER SENSITIVITY [1]

The connected electrical network to be considered consists of N branches, each containing a linear, time invariant resistor, inductor, or capacitor. The extension to nonlinear/time varying networks is trivial. However, existing programs are not usable without modification. Each branch may also contain an independent current and/or voltage source, as well as a dependent current source. The configuration of such a standard branch is shown in Figure 1.

The branch relations for the kth standard branch are

$$E_e(k)^* = E_b(k) + E_s(k) \qquad (1a)$$

$$I_e(k) = I_s(k) + I_b(k) - I_c(k). \qquad (1b)$$

$$k = 1, \ldots, n$$

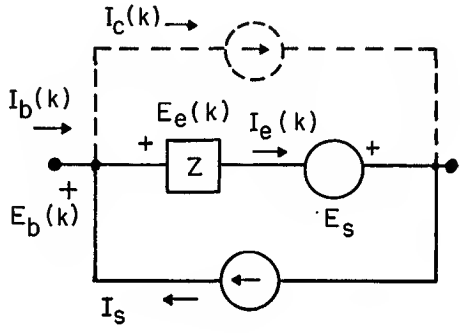


Figure 1.-Standard branch

The element equations for the standard branch will be in one of the following forms:

$$E_e(k) = R(k)\, I_e(k) \qquad (2a)$$

$$E_e(k) = L(k) \frac{dI_e(k)}{dt} \qquad (2b)$$

$$I_e(k) = C(k) \frac{d\,E_e(k)}{dt} \qquad (2c)$$

$$k = 1, \ldots, n$$

In addition, the controlled current source must be considered. It is not an element in the same sense as R(k), C(k) or L(k). The equation for the controlled current is:

$$I_c(k) = \beta_{mk}\, I_e(m) \qquad (2d)$$

In order to find the voltage or current sensitivity of the kth branch with respect to an element variation in the jth branch we take partial derivatives with respect to A(j), the element value of the jth branch.

$$\frac{\partial E_e(k)}{\partial A(j)} = \frac{\partial E_b(k)}{\partial A(j)} + \frac{\partial E_s(k)}{\partial A(j)} \qquad (1a')$$

$$\frac{\partial I_e(k)}{\partial A(j)} = \frac{\partial I_s(k)}{\partial A(j)} + \frac{\partial I_b(k)}{\partial A(j)} - \frac{\partial I_c(k)}{\partial A(j)} \qquad (1b')$$

Rewriting the above equations in more compact form, with derivatives of the independent sources set to zero, we have

$$E'_e(k) = E'_b(k) \tag{3a}$$

$$I'_e(k) = I'_b(k) - I'_c(k) \tag{3b}$$

Note that $I'_c(k)$ will have the form

$$I'_c(k) = \beta_{mk} I'_e(m) \qquad \beta_{mk} \neq A(j) \tag{4a}$$

$$I'_c(j) = A(j) I'_e(m) + I_e(m) \qquad \beta_{mj} = A(j) \tag{4b}$$

Note that the controlled current source is considered part of the branch relations and is not considered an element.

Differentiating the element equations, we obtain

$$E'_e(k) = R(k)\, I'_e(k) \qquad A(j) \neq R(k)$$

$$E'_e(j) = R(j)\, I'_e(j) + I_e(j) \qquad A(j) = R(j)$$

$$E'_e(k) = L(k) \frac{d\, I'_e(k)}{dt} \qquad A(j) \neq L(k)$$

$$E'_e(j) = L(j) \frac{d\, I'_e(j)}{dt} + \frac{d\, I_e(j)}{dt} \qquad A(j) = L(j)$$

$$I'_e(k) = C(k) \frac{d\, E'_e(k)}{dt} \qquad A(j) \neq C(k)$$

$$I'_e(j) = C(j) \frac{d\, E'_e(j)}{dt} + \frac{d\, E_e(j)}{dt} \qquad A(j) = C(j) \tag{5}$$

We may describe the network in matrix form with the following equations

$$[B]E_b] = 0 \tag{6a}$$

$$[S]I_b] = 0 \tag{6b}$$

where [B] and [S] are the circuit and cut-set matrices for the network; E_b and I_b are vectors of branch voltage and branch currents respectively. Differentiating (6) with respect to A(j) we obtain the network sensitivity equations

$$[B]E'_b] = 0 \tag{7a}$$

$$[S]I'_b] = 0 \tag{7b}$$

Substituting (3a) and (3b) into (7a) and (7b), we have

$$[B]E'_e] = 0 \tag{8a}$$

$$[S]I'_e] + [S]I'_c] = 0 \tag{8b}$$

Considering the element sensitivity relations (5) and the topological equations (8a) and (8b), we see that the equations for the sensitivity calculation differ from those describing the original network in three ways:

(1). All independent sources of the original network have been set to zero.

(2). The jth branch now includes a new driving source which may be represented in one of the following ways, depending on the element type ($A(j) \neq \beta_{mj}$):

   a. Resistor: A voltage source of value $I_e(j)$, placed in series with the element R(j)*.

   b. Inductor: A voltage source of value $\frac{d\, I_e(j)}{dt}$, placed in series with the element L(j).

   c. Capacitor: A current source of value $\frac{dE_e(j)}{dt}$, placed in parallel with the element C(j).

(3). If the $\beta_{mj} = A(j)$, a current source of value $I_e(m)$ is placed in parallel with the existing controlled current source.

To apply the above results to a sensitivity analysis, it is convenient to express the voltage driver in the jth branch in terms of its Norton equivalent. Thus, the procedure for calculating network sensitivities to variation in a single component reduces to the following:

1. Construct an "auxiliary" network which is a duplicate of the original network, with all independent sources set to zero (deleted).

2. Drive this "auxiliary" network by means of a dependent current source placed across the jth branch. The source is proportional to the current flowing in the jth branch of the original network if $A(j) \neq \beta_{mj}$. The constant of proportionality is $\frac{1}{A(j)}$ for resistors and inductors, and $\frac{-1}{A(j)}$ for capacitors, where A(j) is the value of the element in the jth branch. For $\beta_{mj} = A(j)$, the current source is $I_e(m)$.

3. The voltages and currents in the "auxiliary" network are respectively the voltage and current sensitivities of the network to a variation of the parameter A(j).

DEVELOPMENT OF MULTIPLE PARAMETER SENSITIVITY CALCULATION [2]

By definition, a reciprocal network has the following property: If an excitation is applied at one pair of terminals in the network, and a response is measured at some second pair of terminals in the network, interchanging the points of excitation and response does not change the ratio between these quantities [3]. Now consider calculation of the sensitivity of the voltage across

branch B with respect to parameter A in branch A. By the reciprocity theorem, the voltage that appeara across branch B when branch A is excited by the current I_A/A will be the same as the voltage that appears across branch A when branch B is excited by the current I_A/A (see Figure 2). Therefore, in the auxiliary network used to calculate the sensitivity function, placing the current source of value I_A/A in parallel with element B rather than in parallel with A, results in the branch voltage across element A being the voltage sensitivity of the voltage across B with respect to element A.

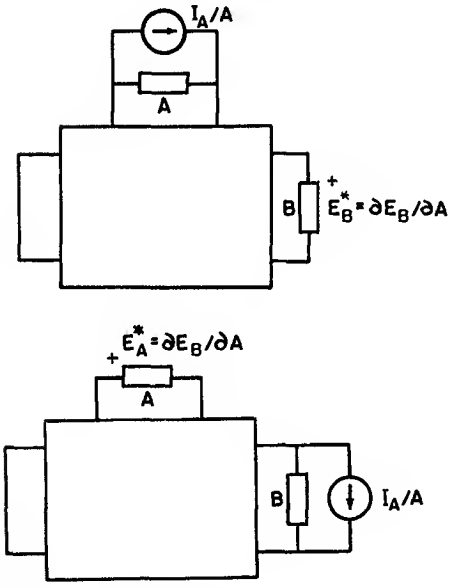


Fig. 2. Property of Reciprocity for Element A in Auxiliary Network

Consider now calculating the A-C sensitivity of branch B with respect to all other parameters, A, B, C, D,....., in the network. Suppose in the auxiliary network, a current source of unity value and zero phase (abbreviated 1/0) is placed in parallel with element B (see Figure 3). Obviously, the voltage E_A^* does not represent the previous voltage sensitivity $\partial E_B/\partial A$ any more, nor is E_C^* equal to $\partial E_B/\partial C$, etc. However, the voltages E_A^*, E_C^*, etc. are related to the sensitivity of E_B with respect to A, C, etc. respectively. If the current had been I_A/A instead of 1, then E_A^* would have been $\partial E_B/\partial A$. Thus, if E_A^* is multiplied by I_A/A, the resulting quantity $(E_A^*)(I_A/A)$ is equal to $\partial E_B/\partial A$.

Similarly, if E_C^* is multiplied by I_C/C, the result is $\partial E_B/\partial C$ (se Figure 3).

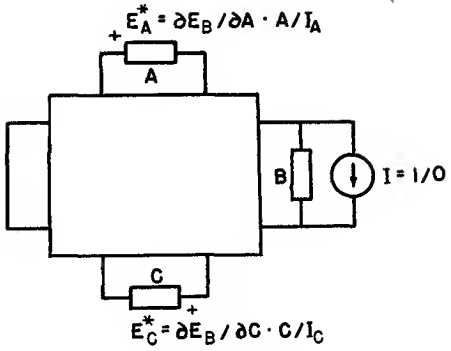


Fig. 3. Auxiliary Network with Unity Value Current Source

Therefore, to obtain the A-C sensitivity of one response voltage with respect to all elements in a reciprocal network, one has the following rules:

1. Place a current source of value unity with zero phase in the auxiliary network in parallel with the element of which the sensitivity of the response function is required.

2. The voltage across any other element in the auxiliary network multiplied by the current through the same element in the original network and divided by the element value itself will give the voltage sensitivity. (-I for capacitors)

3. Repeating (2) with all elements the voltage sensitivity of a single response variable to all elements can be obtained.

EFFICIENT PROGRAMMING OF SENSITIVITY CALCULATIONS

In the case of single parameter sensitivity calculations, the method developed above consists of constructing an auxiliary network and driving this network with a controlled current source proportional to the current in the element whose sensitivity is desired. This interpretation provides the means by which one can use existing programs, such as ECAP or NET 1 without modification. However, it does increase the size of the network which must be calculated. Another way to do the problem is to form the sensitivity equations when the network equations are formed by adding dummy driving current sources to the original network across the elements whose sensitivity is required.

Then, this set of equations is solved with the dummy sources set equal to zero. Each current through the elements whose sensitivity is required is stored. Then, the network is solved recursively with the normal driving sources set equal to zero but the desired sensitivity source not equal to zero. Thus, no network larger than the original network has to be solved. This allows one to use the full capability of the network program to solve a sensitivity problem. This is the method used in RAPID [4].

APPLICATIONS

Much literature exists on the applications of sensitivity calculations. Therefore, two applications of the sensitivity analysis method as presented above, will be given. The applications are a result of the ease that one can calculate multiple parameter sensitivities. The two examples are, (1) the continuously equivalent network and its sensitivities [2] and (2) evaluation of mathematical models [5].

The continuously equivalent network is a network in which the elements are varied as a continuous function of some dummy variable, but whose specified properties remain constant. By defining a performance index which is a function of the sensitivity function, the "best" continuously equivalent network can be selected out of all possible ones. Previous work examined this problem at one frequency [6]. With the new technique, these networks were examined over a range of frequencies. The experimental results of this study allows one to make three hypotheses about continuously equivalent networks:

1. The continuously equivalent network resulting from a minimization of the sum of the magnitudes squared of the sensitivity functions at a given frequency is the network with minimum sum of the msgnitudes squared of the sensitivity functions at all frequencies.

2. The sum of the magnitudes squared of the sensitivity functions decreases as the number of elements increases in continuously equivalent networks.

3. The sum of the sensitivity functions is invariant with respect to the various equivalent networks.

The most important hypothesis is the first. If true for all continuously equivalent networks, then a great simplification of the computation problem results. Even if a limited class of continuously equivalent networks exhibit this property, then the result is worthwhile. These experimental results do not prove that all continuously equivalent networks have these three properties, but they do provide a challenge for future work. At least they indicate a class of continuously equivalent networks which have this property.

In recent years several investigators have constructed models to represent the response of the human cochlea [7] [8]. These models are partial differential equations which, in one case at least, have been reduced to an electrical analog. As an example of what could be done with sensitivity calculations, the electrical analog system was solved using ECAP. Then, using the method of multiple sensitivity analysis, the response of the network was calculated and compared with experimental measurements. Then, a linear programming optimization routine was used to adjust the parameters in order that the error between the model and the experimental measurements was minimized. The result was that the experimental measurements and model agree quite closely in magnitude, but not in phase. It is conjectured then, the errors in phase are not due to model errors but are due to measurement errors [6]. These results will be presented in detail elsewhere.

ACKNOWLEDGEMENT

This work was supported in part by NSF Grant GK-262 and NSF Grant GU-1153.

REFERENCES

[1] J. V. Leeds, "Transient and Steady-State Sensitivity Functions", IEEE Transactions on Circuit Theory, CT-13-3, September 1966, pp. 288-289.

[2] J. V. Leeds, Gabor Ugron, "Simplified Multiple Parameter Sensitivity Calculation and Continuously Equivalent Networks", June 1966, EE-66-3, Rice University, Department of Electrical Engineering (Accepted IEEE CT).

[3] S. Seshu-N. Balabanian, Linear Network Analysis, John Wiley and Sons, Inc. New York, 1963, pp. 160-162.

[4] J. V. Leeds, G. L. Barksdale, G. Gruenreich, C. M. Moore, "Computer Analysis and Design of Networks", Proceedings of IEEE, Vol. 45, No. 12, p. 1892.

[5] Talbot S. Huff, Jr., "An Optimized Model of The Human Cochlea", Master of Science thesis, Rice University, June 1967.

[6] J. D. Schoeffler, "The Synthesis of Minimum Sensitivity Networks", IEEE Trans. CT-11, June, 1964, pp. 271-276.

[7] Ewald Glaesser, An Analog of the Ear (unpublished doctoral dissertation, University of Arizona, Tucson, 1962).

[8] C. Wansdronk, "On the Mechanism of Hearing", Phillips Research Report Supplements, I (1962).

CIRCUS, A DIGITAL COMPUTER PROGRAM FOR
TRANSIENT ANALYSIS OF ELECTRONIC CIRCUITS

By

RICHARD H. DICKHAUT

Mr. Dickhaut received a B. S. degree in Physics from the University of Massachusetts in 1956, and did graduate work in Mathematics and Physics at Northeastern University and the University of Washington.

He joined Boeing in 1959, and has worked continuously in the field of radiation effects, developing a radiation effects data reduction computer code and the CIRCUS transient analysis code, as well as conducting experimental studies on semiconductor components and circuits, and designing radiation-hardened circuits.

Mr. Dickhaut is Head of the System Radiation Hardening Organization and has been author or co-author of 12 technical papers in the field of radiation effects.

CIRCUS, A DIGITAL COMPUTER PROGRAM FOR TRANSIENT ANALYSIS OF ELECTRONIC CIRCUITS

By Richard H. Dickhaut

Research Engineer
The Boeing Company
Seattle, Washington

SUMMARY

CIRCUS is a digital computer program which simulates the time domain response of an electronic circuit to an arbitrary forcing function. Although motivation for the development of CIRCUS originated from efforts to evaluate transient nuclear radiation effects on electronics, the program is applicable to the much wider class of everyday electronic design problems, and any physical system having an electrical analogue.

Like other time domain circuit analysis programs such as NET-1, PREDICT, and SCEPTRE, CIRCUS requires a topological description of the circuit which specifies element values and forcing functions, and from this information constructs the appropriate time-domain equations, finds the dc initial conditions, and then evaluates the transient solution by numerical integration of the differential equations. Significant features of the program are its speed, flexibility, easy input and output programming, and present availability on four different computer systems.

Introduction

In the past few years, the spurious effects generated by transient nuclear radiation on electronic systems have come under intensive study by designers of military electronic systems. Although the radiation pulse incident upon a given electronic system is transient, the effects it causes in the electronics may be both transient and permanent, and these effects may lead to either transient or permanent system failure. To analyze such effects, it was evident that some means of establishing the vulnerability of electronics to radiation without extensive and costly system testing was required. The attempt to analyze complex circuitry by hand presented considerable difficulties. Eventually, digital computer aids were selected as the most practical way to solve the equations which describe a circuit. Thus, computer programs for time domain circuit analysis have been among the important tools which have been developed to study and, ultimately, to control the effects of radiation on electronic circuits. CIRCUS was developed to meet such a need, and has been used extensively for the past three years in simulation, prediction, and radiation hardened design of complex circuits exposed to various radiation environments.

Although motivation for the development of CIRCUS originated from efforts to evaluate transient nuclear effects, the program is applicable to the much wider class of everyday electronic design problems, using a variety of different electrical forcing functions. In addition, since most physical systems have an electrical analogue, it may be extremely useful for investigating the transient behavior of mechanical, hydraulic, physiological, and other systems.

This paper will present a brief description of some of the CIRCUS features of particular interest to the user.

CIRCUS Features

CIRCUS (Circuit Simulator) is capable of analyzing any arbitrary circuit made up of resistors, capacitors, inductors, semiconductor devices, and voltage and current sources. As in other programs of this type, special efforts have been made in the design of CIRCUS to ensure ease of use and simplicity in the input required by the programmer.

Like other time-domain circuit analysis programs such as NET-1, PREDICT, and SCEPTRE, CIRCUS requires a topological description of the circuit which specifies element values and forcing functions, and from this information constructs the appropriate time-domain equations, finds the dc initial conditions, and then evaluates the transient solution by numerical integration of the differential equations. Each program offers its special features, adaptability to given computer types and inherent advantages and disadvantages. It is assumed that the circuit designer wishing to use computer aids has in mind certain requirements for program features and flexibility when he chooses a given computer program for his specific needs. A complete description of CIRCUS is given in References 1 and 2 and from a previous study was shown to be significantly faster than previous programs (Reference 3). In general CIRCUS performs the same type of circuit analysis as SCEPTRE, with special care taken to provide many user-oriented features. However, SCEPTRE has features, if needed, which make it a desirable tool for the development of equivalent circuits. A brief summary of some of the more important features of CIRCUS are the following:

1. Stored Semiconductor Models and Device Parameter Library -- CIRCUS has stored device models for conventional and field-effect transistors; conventional, tunnel, and zener diodes; and 4-region devices. For each of these models, there are built-in photocurrent generators for radiation effects work. Of course, these models, in combination with other elements, can give rise to other equivalent circuits of interest. One has only to specify the terminal connections of a given device and either (a) request the program to obtain the device data from the device parameter library tape, or (b) supply the appropriate device parameters as part of the input, or some combination of both procedures.

The stored models are based on a charge-control description of semiconductor device operation. These models in CIRCUS have been transformed into current rather than charge parameters and appear in an equivalent T-circuit very similar to the Ebers-Moll description. In fact, simple algebraic conversion can be made between charge-control parameters and Ebers-Moll parameters as described in (1), and this procedure has been used to demonstrate that identical circuits analyzed on both NET-1 and CIRCUS with their respective stored models give an identical transient analysis (Reference 3). Both these models ultimately reduce to the same differential equations, and accurately describe both large and small signal operations in all four regions: cutoff, active-normal, active-inverted, and saturation. Storage effects are included.

2. Special Purpose Parameter Variations and Multiple Analyses -- CIRCUS allows, in one computer run, multiple analyses of a given topology circuit with arbitrary changes in any or all element values, or analyses of many circuits with different topologies, CIRCUS also will allow a change in parameters (on a restart mode) of any given analysis, as well as at any predetermined time during an analysis. These latter capabilities have many uses, but among those most obvious are: (a) changes in parameters affecting short-time constants in low frequency circuitry after fast transients have passed, allowing for more rapid calculation; (b) degradation of device performance at a given time due to nuclear radiation; and (c) changes in forcing functions of circuits in AC equilibrium.

3. Output Options -- A variety of options are available as output for CIRCUS. In addition to any or all node voltages, and terminal currents for any elements, the user may request (1) internal currents and junction voltages internal to any semiconductor device, (2) a monitor on voltage, current, or power dissipation in any element, (3) voltage across resistors, capacitors, inductors, and current sources, (4) the sum or difference of any two node voltages, or (5) any variable minus its DC steady state pulse. Only those variables selected by the user will be printed and/or plotted for display, and all information is displayed at selected, variable intervals chosen by the user. These output options allow the designer to monitor any desired variable, and to perform optimization studies. They allow the circuit or system analyst to identify failure modes easily.

4. Exponential Method of Numerical Integration -- CIRCUS solves the network differential equations for the transient analysis by using a variation of an exponential method developed by D. A. Pope (Reference 4). As programmed into CIRCUS, this integration routine is a one-pass, variable-step method controlled both by the solution behavior and by the smallest local time constant of the network. The fundamental step-size control parameter is related to the number of terms required for convergence. Experience with a wide variety of circuits has shown that this exponential method uses a step-size which is from three to four times as large as in the fourth-order variable-step Runge-Kutta method. The machine time per step is about the same in both methods since the time spent in the three extra derivative evaluations per Runge-Kutta step is approximately the same as the time spent by the exponential method in computing the exponential of a matrix. Therefore, due to the larger step size, the exponential method will evaluate a given transient solution in one-fourth to one-third the time taken by a fourth-order Runge-Kutta method.

5. Machine Independence -- CIRCUS has been coded almost entirely (98%) in Fortran IV and thus can be used on any computer which has a Fortran IV compiler and adequate storage. CIRCUS has already been coded for use on the following machines at several installations in the country: IBM 7090/7094, UNIVAC 1107/1108, G.E. 625/635, and the CDC 6600. It is expected that an IBM 360 version will be available in March of 1967.

Conclusion

CIRCUS in its present form already represents a flexible and efficient tool for computer-aided design. It is:

1. user-oriented rather than programmer-oriented, meaning that its special features (multiple reruns with changed parameters, restart mode, stored models, etc.) are designed so that the electronic engineer or vulnerability analyst will find it easy to use.

2. computationally fast, and thus an economical tool as circuit analysis programs go.

3. a proven and debugged program as attested by three years of uninterrupted use.

4. relatively machine-independent, and has already been coded for use on four different computer systems.

Ideas for further development of the program include new component models and more

computational flexibility. One specific computational goal is to provide the capability for fast, efficient analysis of low frequency circuits having high frequency components. It is expected that resulting modifications to the program will significantly enhance the possibilities of computer-aided design, with the ultimate goal of increasing the productivity of the design engineer, and improving the performance and reliability of the final product at a significant reduction in cost and time.

References

1. L. D. Milliman, W. A. Massena, and R. H. Dickhaut, "CIRCUS, A Digital Computer Program for Transient Analysis of Electronic Circuits -- User's Guide, Harry Diamond Laboratories, 346-1", January, 1967.

2. L. D. Milliman, W. A. Massena, and R. H. Dickhaut, "CIRCUS, A Digital Computer Program for Transient Analysis of Electronic Circuits -- Programmer's Guide, Harry Diamond Laboratories, 346-2", January, 1967.

3. R. H. Dickhaut, "Comparison of Three Computer-Aides Design Programs", Electro-Technology, Vol. 79, No. 1, pg. 88, January, 1967.

4. D. A. Pope, "An Exponential Method of Numerical Integration of Ordinary Differential Equations", Communications of the ACM, Vol. 6, No. 8, pp. 491-493, August, 1963.

TOPOGRAPHIC SIMULATION AS AN AID TO PRINTED CIRCUIT BOARD DESIGN

By

CLIFFORD J. FISK

Mr. Fisk graduated from the University of Utah in 1950 with a B. S. E. E. degree. He has been employed at Sandia since 1958. He is currently a staff member in the Advanced Techniques Division of the Sandia Programming Department.

LESLIE E. WEST

Mr. West received his B. S. E. E. degree from the University of New Mexico in 1945. He was employed by Los Alamos Scientific Laboratory from 1946 to 1949 and by Sandia Corporation from 1949 to the present. He has been in the Computer Programming group for the past 11 years and is currently in the Advanced Techniques Division.

DAVID CASKEY

Mr. Caskey received his B. S. E. E. and M. S. E. E. degrees from the Massachusetts Institute of Technology in February, 1964, and has been employed by Sandia Corporation since July of that year. He has been with the Advanced Techniques Division of the Programming Department there for the past year.

TOPOGRAPHIC SIMULATION AS AN AID TO PRINTED CIRCUIT BOARD DESIGN

By L. E. West, 9424 and D. L. Caskey, 9424

Sandia Laboratory, Albuquerque, N. M.

SUMMARY

A new method has been programmed to produce the conductor path layout as part of the continuing development of the ACCEL system. The method is based on a topographic simulation scheme where component connector pin locations are represented by peaks, and the areas between pins are represented by combinations of slopes, plains and valleys. Examination of this terrain can lead to the establishment of trails to represent conductor paths. Also, methods of displaying the simulated topography and trails have been programmed. Although this presentation is centered around the circuit board problem, the technique may be useful in other problems of a contextual nature.

Introduction

The topographic simulation technique, presented here, is part of the continuing development of the ACCEL[1] system. It is based on the idea that the configurations of a circuit board can be represented in a topographic structure. This idea was first introduced to the authors by Dr. Iben Browning, Executive Director of the Thomas-Bede Foundation in Los Altos, California.

Topographic simulation, per se, is not new; it has been used in physical models to represent interrelated, steady-state forces.[2] However, it is believed that the sense in which it is used here is new; i.e., combining in the computer a topographic scheme with self-modifying characteristics in such a way as to produce a dynamic topography. As used here, this dynamic property is especially vital in representing the constantly changing conditions caused by the growth of circuit paths. Another unique feature allows any given point to be "negative" in relation to all other points; i.e., everything is downhill from everything else. So, in essence, we have a multi-dimensional, dynamic topography.

Topographic Structure

In the computer, the topographic area is represented by an array whose dimensions are 100 x 100 elements (this size is limited only by the available core storage in the computer). The location of each element represents a corresponding location on the circuit board, scaled on the 100 x 100 grid. Each element is a computer word and contains a number representing the "altitude" of its location.

The topographic structure that will be created within this framework represents physical features occurring on the circuit board. At the beginning of the process, the following three features can be represented:

1. The shape (outline) of the circuit board
2. Holes or other obstacles to conductor paths
3. Component and connector pin locations (lands).

The conductor paths, plated-through holes, and certain trouble spots that are established by the process are added to the topographic structure as they occur.

The process begins with all elements being set at zero altitude. The data, representing electrical contact locations (component pins) and the locations of physical obstacles (such as holes in the board), are fed into the system. A "peak" is created in the grid at each such location. That is, the grid element representing a pin location is given a high altitude, say 1000; the elements surrounding it are given a lower altitude, say 750; and the next ring of elements still a lower altitude, etc. Then, to represent the edges of the circuit board, a "ridge" is created around the periphery of the grid. This ridge slopes inward from each of the four edges, forming a basin that contains the peaks. The actual form of the peaks and the basin is variable, being governed by control data used in the equation described below.

Although there are several ways in which the structure may be numerically developed, the one used most extensively in the current study is of the following form:

$$A = \frac{K}{C \times D + 1}$$

where A is the altitude of any given element,

C is a constant that controls the slope of the basin and peaks,

D is the distance (on the grid) between the element and the nearest edge of the grid or between the element and the center location of the nearest peaks, and

K is the constant that determines the magnitude of the altitude.

Additional information can be represented by replacing K with a function. For example, if a peak represents a high-voltage contact on the board, K could be a function of voltage and thus cause low-voltage paths to avoid that area. Or, similarly, if a location in the circuit is particularly sensitive to the capacitance effects of conductive paths, the K (for peaks in that vicinity) could be a function of sensitivity and thereby provide better dispersion of conductors.

The variety of conditions that can be accommodated by this topographic structure makes it an especially powerful technique for the problem at hand and, perhaps, for other problems that involve the interrelationships of many conditions.

After the basin and peaks have been formed, the process of establishing circuit paths begins. That process is discussed in the following section of this report. However, changes in the topography occur during the process and should be mentioned here in order to complete the description of this subject. These changes do not occur in any particular sequence but, instead, occur as the requirement for them is determined by the system. The following conditions cause the topography to be modified:

1. As the locations of circuit paths are established, ridges are built along the paths. This, in effect, forms valleys between completed paths. The purpose is to help the uncompleted paths find their way around or between completed paths.

2. When a path runs into a trouble spot in the terrain, it tends to wander in a tight zigzag fashion, trying to find a way out. Trouble spots may be caused by conditions such as low areas surrounded by peaks, by very narrow valleys caused by ridges of several completed paths, or by a V-shaped or U-shaped bend in a completed path. If a path gets trapped in such an area, the altitude of that vicinity is progressively increased. This causes the path to modify its direction and, hopefully, find a way out. It also helps other uncompleted paths to avoid that area.

Up to this point, the topography that has been described is somewhat comparable to natural topography. This comparison is at least close enough that features occurring in natural topography, such as ridges and valleys and peaks, are useful in describing the imaginary topography in this system. The one remaining topographic characteristic, yet to be described, has no natural counterpart, and therefore, must be imagined in a different way.

Perhaps the best way to view this feature is to imagine that the topography described thus far is built on a rubber sheet that is stretched on a frame. Then when a step is being taken (a step representing an increment of a circuit path) from some location toward a target location, the rubber sheet is pressed down at this point of the target location, forming a cone with the ridges, peaks, etc., on its inside surface. When the step has been taken, its target location is released and the system is prepared to find the next step. An iterative step-sequencing system is used (that is, commutation of all of the uncompleted path ends) which gives the effect of developing all paths simultaneously. Therefore, the next step will probably be with an increment, a path, and a target different from those in the preceding step. The new target location is pressed down, a step is taken, and the target is released. This process of pressing down and releasing successive target locations is an integral part of the procedure for establishing circuit paths. Its exact purpose will become more evident in the following section, where the method of evaluating a path step is discussed.

The following figures, produced by Fortran programs processed on the IBM 7094 and plotted by the SC4020 plotter, help the reader to visualize the simulated topography. Figure 1 corresponds to a contour map showing the basin and peaks at the beginning of the process. Figures 2 through 5 correspond to relief maps. Figure 2, like Figure 1, shows the basin and peaks; Figure 3 represents the same structure, except with one of the peaks pushed down to form a target cone. Figure 4 contains the same set of peaks, but is later in the process after ridges have been built to represent completed paths. Figure 5 shows the effect of a target cone at this stage.

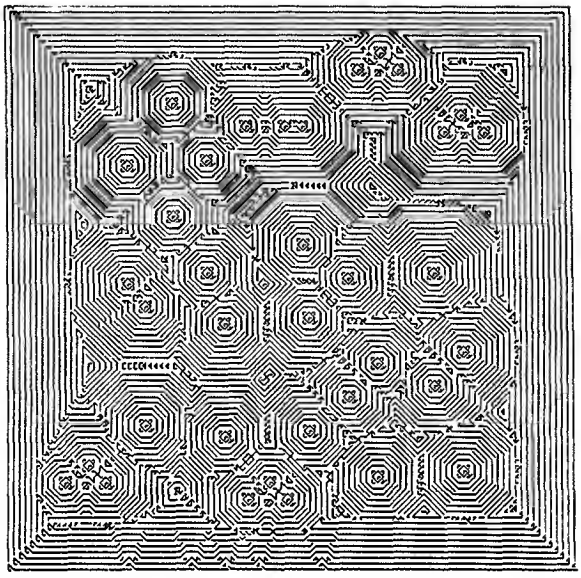


Figure 1. Contour Map, Showing Basin and Peaks at Beginning of Process

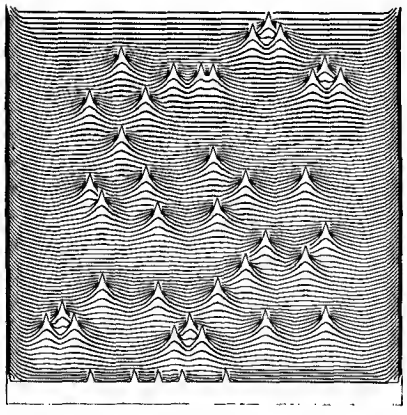


Figure 2. Basin and Peak Structure at Beginning of Process

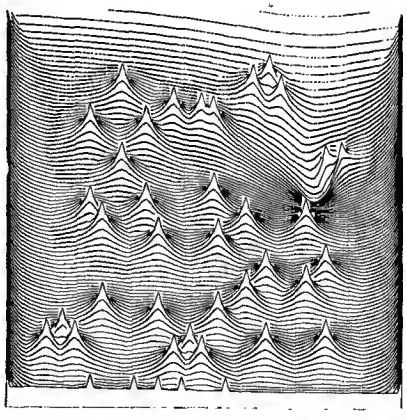


Figure 3. Same Structure, With One Peak Pushed Down to Form Target Cone

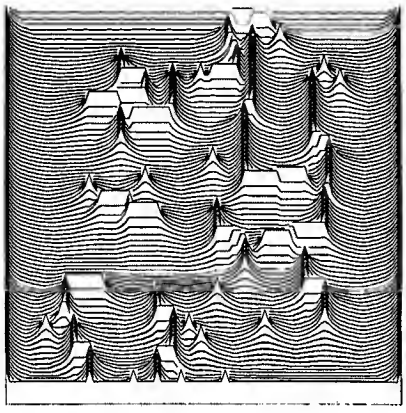


Figure 4. Same Structure, With Ridges

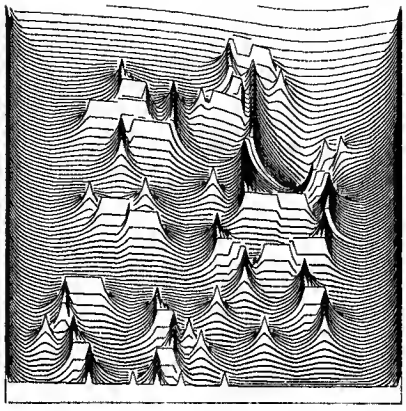


Figure 5. Effect of Target Cone at Later Stage of Process

Algorithm for Path-Step Selection

The system contains two general features that work together in the just-described topography to establish the location of circuit paths. One is the basic algorithm for step sequencing and step sequencing and step evaluation described below. The other comprises the usage and variations of this algorithm in the several phases of the system which are described in the following section.

The definition of a path step, as used here, is an element in the 100 x 100 grid and represents a portion of a conductor path. If a path is to be developed between peaks A and B, it will be composed of at least as many steps as there are elements in the grid between A and B. Since paths frequently need to wander (for example, around other peaks), the number of steps to complete a path depends upon the obstacles between A and B.

The step-selection procedure is basically one of examining the slope of the terrain in the immediate vicinity of the element from which a step is to be taken and choosing the direction that is most downhill. The exceptions to this rule are that a step must not be on or adjacent to steps of a different node, and a step cannot go backward (i.e., a path cannot intersect itself). Because of these exceptions, the most downhill direction may be unacceptable. Therefore, it is necessary to provide alternate choices of step direction. Another restriction, which is purely for the sake of simplicity, is that a step must be one of the four orthogonal elements of the current location (i.e., it cannot step to a diagonal element).

An interesting feature of the stepping algorithm is that the two active ends of a path try to find each other. This stepping is controlled by a data array that keeps track of the most recent step location of each path end, as well as the point of origin (peak) of each path end. For definition, the most recent step of a path end is called the "target" of the other end: the peak toward which a path is moving is called the "target peak" (i.e., the point of origin of the target path end); and the path end that is in the process of taking a step is called the "active end." During the step-evaluation process, three characteristics of altitude are measured for each of the orthogonal elements at the active end:

1. The actual altitude currently in the topography.
2. The portion of item 1 which is due to the target peak, visualized as though the target peak had a cumulative effect upon the entire terrain.
3. The effect on altitude caused by the target being pulled down to form a cone, as described earlier.

Items 2 and 3 are computed values produced by the following equations:

$$\text{Altitude due to Target Peak} = \frac{1000}{C \times D + 1}$$

where C is a constant that defines the shape of the peak, and D is the distance (grid elements) between the center of the peak and the element being considered.

> Altitude Change due to Cone = K x (100 - D) where K is a constant that defines the depth of the cone, and
> D is the distance from the target to the element being considered.

Then, for each of the orthogonal elements, an effective altitude is computed as being item 1 minus item 2 minus item 3. Whichever of these elements has the lowest effective altitude and does not violate any of the restrictions described earlier will be the new step location.

Process Phases -- The overall operation of the system is divided into several phases. Each phase, while employing the algorithm previously discussed, follows a slightly different policy in making connections. At the present time, the system employs four separate phases. A phase control subroutine is used to oversee the sequence of phases. It is easily varied to try different policies for making connections.

Front of Board--Phase 1 -- Phase 1 begins with a topography that includes only the border and the peaks at the land locations. During this phase, steps are taken from both ends of a connection pair toward each other. A step from both ends of all connection pairs constitutes one cycle of the stepping algorithm. It is important to note that during this phase there are no ridges to represent paths in the terrain. However, a step is prohibited if its location would be adjacent to a previous step from a wire on a separate node. Thus there is an infinite barrier surrounding the wires as they progress, even though the sloping ridges are not present. Consequently, no early warning is provided to warn of an impending obstruction. After the cycle limit is reached or all the wires become trapped, whichever comes first, Phase 1 terminates. Typically, about half the connections are made during Phase 1.

Phase 2 -- Phase 2 is almost a repeat of Phase 1, but with one important difference. All the wires completed in Phase 1 are provided with ridges, sloping downward and outward from both sides of the connection path. Of course, all those connections not completed are wiped out, and Phase 2 attempts them once more, but now with more contextual information from which to work.

Phase 2 is the last attempt to make the connections on the front of the board. Therefore, during this phase, a special routine is used to determine whether the steps from the two ends of the wires are still making progress toward each other. If this routine determines that one end is beginning to stray, its progress is halted and a land is placed there. Thus, when Phase 2 terminates, for each wire that has still not completed a connection, a land is placed at the last position of the two ends.

If a two-sided board is being generated, the program now tries to complete the remaining connections, between the new land locations, on the back of the board. The completed path locations that have been established for the front side of the board are saved and the topography is initialized to zero before beginning on the back side.

Back of the Board--Phases 1 and 2 -- The topography representing the back of the board is developed, placing the border as before, and creating the peaks as before. However, there will be many more lands or peaks than there were for the front of the board. The newly generated lands at the ends of uncompleted wires, as well as the original lands, make up the data for calculating peaks. (These land positions must be avoided because the lands protrude through the circuit board.)

After setting up the terrain for the back of the board, Phase 1 is utilized as before. Of course, only those wires still uncompleted are attempted, and they step from new land locations.

Again, after Phase 1 is completed, dikes are generated for newly completed paths, and uncompleted paths are erased. Phase 2 is then tried. After Phase 2, if there are still uncompleted paths, Phases 3 and 4 are tried.

Phases 3 and 4 -- Phases 3 and 4 differ from Phases 1 and 2 in two ways: (1) only one wire is tried at a time and (2) steps are taken from only one end of the wire. Thus, these phases are executed once for each uncompleted wire. Phase 3 is tried first. During this phase, end 1 of the wire is trapped. In other words, it stays at its land location while only end 2 steps toward that opposite land. It is important to note that during this phase steps are taken for only one wire path.

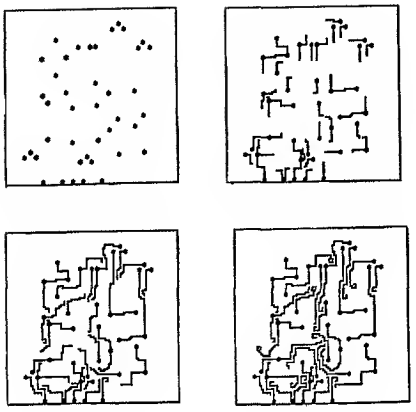


Figure 6. Stepping Sequence, Phase 1

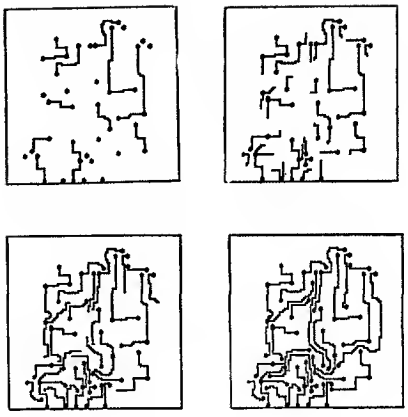


Figure 7. Stepping Sequence, Phase 2

After all wires have been tried, Phase 4 is entered. It is identical to Phase 3, except that the remaining uncompleted wires are tried from the other end. That is, end 2 is trapped, while end 1 does the stepping. After all wires that are still uncompleted have been attempted by Phase 4, the program generates a plot tape for producing a picture of the board and conductor paths and then goes on to the next circuit to be routed.

The following figures illustrate the progression of paths in the terrain shown in Figures 1 through 5. Figures 6 and 7 show the progress of paths in Phases 1 and 2, respectively. Note that the first frame of Phase 2 contains those paths that were completed in Phase 1.

The following figures illustrate the final product of the system, using both sides of the board. Figure 8 is a diagram of a particular circuit, showing the placement of components and the connections that need to be made. Figures 9 and 10 are plots, produced by the system, to provide the required connector paths. Figure 9 represents the front side of the board, and Figure 10 the back. In this particular case, all connections are completed.

The component placement shown in Figure 8 was computer generated, using the current ACCEL system.

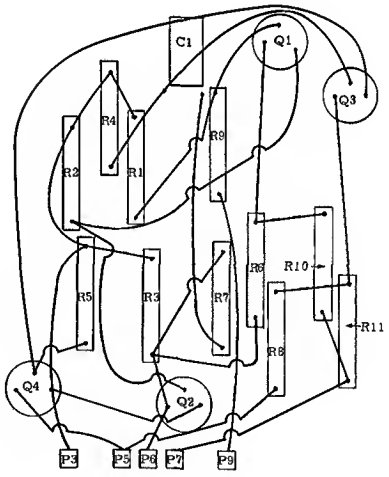


Figure 8. Diagram of Circuit, Showing Placement of Components and of Connections to be Made

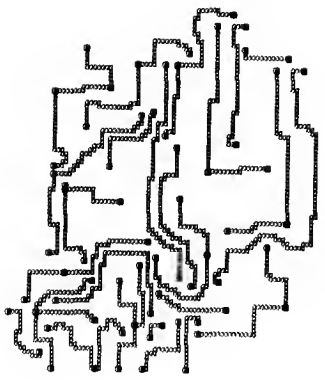


Figure 9. Plots Providing Required Connector Paths (front of board)

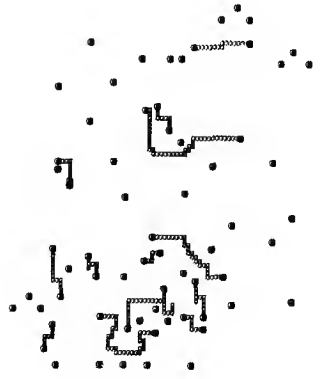


Figure 10. Plots Providing Required Connector Paths (back of board)

Conclusions -- Although it is somewhat premature to compare the performance of this technique to the routing technique used in the operating ACCEL system, comparative results from two test circuits reveal that, in terms of the number of connections completed, the topographic system did slightly better in both tests. Also, the arrangement and dispersion of paths appear to be better. On the other hand, the topographic system left a few paths uncompleted that the ACCEL system would have completed with ease. Therefore, a combination of the two techniques seems most promising. Since electrical characteristics (aside from connection requirements) can be represented in the topography, the system has distinct appeal over analytical methods that only consider minimizing crossovers.

References

1. Fisk, C. J., and Isett, D. D., Automated Circuit Card Etching Layout, SC-TM-65-544, October 1965.

2. Zworykin, V. K., and Morton, G. A. Television, the Electronics of Image Transmission in Color and Monochrome, 2nd. ed., J. Wiley, 1958, p. 108.

This work was supported by the United States Atomic Energy Commission.

SCEPTRE: A SECOND GENERATION TRANSIENT ANALYSIS PROGRAM

By

STEPHEN R. SEDORE

Mr. Sedore received his B. S. E. E. degree from Indiana Institute of Technology in 1958, and his M. S. E. E. degree from Syracuse University in 1967. His Master's thesis was concerned with the mathematical formulation of an automatic transient analysis program.

Mr. Sedore was involved in transistorized circuit design for the first five years of his career, and has been involved in the formulation and use of automatic circuit analysis programs for the past three years.

SCEPTRE: A SECOND GENERATION TRANSIENT ANALYSIS PROGRAM*

By Stephen R. Sedore

Staff Engineer
International Business Machines Corporation
Federal Systems Division
Electronics Systems Center
Owego, New York

SUMMARY

This paper presents a balanced discussion of SCEPTRE (System for Circuit Evaluation and Prediction of Transient Radiation Effects) from both the academic and practical points of view. Two basic incidence matrices that apply to any connected electrical network composed of resistances, capacitances, inductances, voltage sources and current sources will be used to derive a general F matrix. The F matrix will be used in turn to establish very practical relations between the tree branches and links of the general electrical network. It will be shown that this relationship is exploited to form an efficient algorithm that is used in forming some of the equations that are required for a general transient solution.

The second part of this paper will describe the most important features that are available to aid in the practical solution of contemporary networks. Each of these features were designed to provide more convenience to the user without compromising the overall flexibility of the program.

INTRODUCTION

Automatic circuit analysis programs are intended to relieve the user of the tedious and error-prone tasks of writing and programming circuit equations. As a result, the user is only required to entsr a topological and quantitative description of each network of interest, according to an easily learned format. The general acceptance of programs of this type can be gauged by the steadily increasing volume of reports and descriptions that are currently appearing in many technical journals.

In 1963, a group at IBM's Electronics Systems Center was funded by the Air Force Weapons Laboratory at Kirtland Air Force Base, New Mexico, to develop an automatic circuit analysis program. This program was to be flexible enough to accommodate a variety of non-conventional effects that are of interest in transient radiation problems ae well as general enough to handle conventional transient problems. The result was PREDICT (Prediction of Radiation Effects Using Digital Computer Techniques), an automatic transient program that has been in general dissemination throughout the country since 1964. Inevitably, the need for a successor program (SCEPTRE) slowly became evident. Invaluable comments, pro and con, flowed in from the more activs and experienced users. This information played an important part in the development of many of the features that today distinguish SCEPTRE from PREDICT. The reader who is seriously interested in the progress that has been made in this field would do well to rsad and compare the respective manuals for these programs (PREDICT [1], SCEPTRE [2]). Dissemination of both programs is controlled by Lt. Gary Pritchard, Air Force Weapons Laboratory, Kirtland AFB, New Mexico 87117.

*The work described in this paper was sponsored by the Air Force Weapons Laboratory under Contract AF29(601)-6852, Supplemental Agreement 1, as part of the Defense Atomic Support Agency program under NWER, Subtask 16.015.

Symbols

Y	vector of state variables
$\dot{Y}$	vector of state variable derivatives
b	number of network elements; i.e., sum of all resistors, capacitors, inductors, voltage sources and current sources
n	number of network nodes
I_b	vector of currents through all network elements
V_b	vector of voltages across all network elements

F — an incidence matrix that expresses a relation between network links and tree branches

F^T — the transpose of F

The description of a SCEPTRE formulation, must first begin with the definition of a tree [3]. A tree is a connected subgraph of a network which includee every node in that network, but contains no closed loops or meshes. A specific tree may be defined as one in which elements are inserted in the following order of priority: voltage sources, capacitors, resistors, and inductors. All current sources are excluded from this tree. All elements that are included in the tree are called tree branches; all elements that are excluded from the tree are called links.

The state variables of any system may be defined as the minimum set of variables which, together with all inputs, are sufficient to determine all other system quantities at any particular instant of time. All system quantities at time t_n are dependent upon the values of the state variables and inputs that hold at time t_n and are not directly dependant on those at any other time. It can be shown that the state variables of the general electrical network are the set of capacitor tree branch voltages and inductor link currents. This choice of state variablss is equivalent to specifying the independent initial conditions of the network. It may be stated further, that the choice of state variables, leads to a set of simultaneous first-order differential equations [4].

The solution process is best illustrated with the highly simplified block diagram of Figure 1. The process begins with the insertion of the state variables Y that are valid at time t_n where n=0 at the start of a transient problem.[1] All currents, voltages and state variable derivatives that are valid at $t=t_n$ are computed and output (if desired). The state variable dsrivatives $\dot{Y}(t_n)$ are numerically integrated by the integration routine to provide the state variables $Y(t_{n+1})$ that are valid at the next solution increment. It is worth noting that the block diagram serves to illustrato the two most basic factors concerned with the amount of computer time required for the solution of transient problems. The number of solution steps required to complete a given problem depends on the integration routine used, and the amount of computer time required per solution stsp depends on the efficiency with which the algebraic operations are carried out in the Equation Solution block.

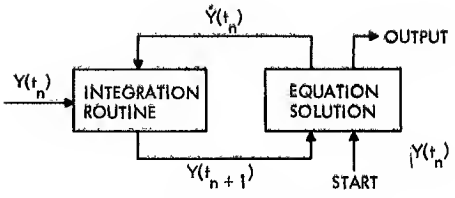


Figure 1. Solution Process Diagram

The topological makeup of any connected network may be described by means of two incidence matrices. The first of thess is the fundamental cut set matrix Q where:

$Q = [q_{ij}]$ is a matrix containing n-1 rows and b columns for the general connectsd network consisting of n nodes and b elements where:

q_{ij} = +1 if the direction of fundamental cut set i coinoidss with the reference direction of element j in fundamental cut-set i

q_{ij} = -1 if the dirsction of fundamental cut-set i is in opposition with the rsfsrencs direction of element j in fundamsntal cut-set i

q_{ij} = 0 if the fundamental cut-set i does not include element j

The direction of each fundamental cut set is customarily chosen to agree with that of its associated tree branch. It will be convenient to partition Q according to links and tree branches as

(1) $$Q = [Q_L \; Q_T]$$

which, if ths rows and columns are ordered appropriately, becomes

$$Q = [-F^T \; U]$$

where the (n-1) rows and columns of the unit matrix U corrssgpond to the tree branches and the (b-n+1) columns of $-F^T$ correspond to the links associated with any tree.

Another incidence matrix can be defined which reprssents the topologieal relationship betwsen the elements and the meshee, or loops, or circuits, of the

1. The initial values of the state variablee may be eupplied, if known, by the user. Otherwise, they may be automatically determined by an iterative procedure that is contained in SCEPTRE.

network. This matrix will be called the fundamental circuit matrix T, where

$T = [t_{ij}]$ is a matrix containing (b-n+1) rows and b columns for the general connected network consisting of n nodes and b elements where:

t_{ij} = +1 if element j is part of fundamental circuit i, and if the reference directions of element j and fundamental circuit i coincide.

t_{ij} = -1 if element j is part of fundamental circuit i, and if the reference directions of element j and fundamental circuit i do not coincide

t_{ij} = 0 if element j is not part of fundamental circuit i

The direction of each fundamental circuit is customarily chosen to agree with that of its associated link. The number of rows in the T matrix corresponds to the fact that only (b-n+1) circuit equations are required for the solution of any network. If the columns of T are ordered appropriately, T may be partitioned as

$$(2) \quad T = \begin{bmatrix} U & F \end{bmatrix}$$

where, the (b-n+1) columns of the unit matrix U correspond to the links and the n-1 columns of F correspond to the tree branches of any tree. If the columns of Q and T are arranged in the same element order, it is well known that:

$$(3) \quad Q\ T^T = 0$$

Also, since

$$(4) \quad Q\ I_b = 0$$

it always true, a simple partition permits

$$(5) \quad \begin{bmatrix} -F^T & U \end{bmatrix} \begin{bmatrix} I_L \\ I_T \end{bmatrix} = 0$$

leading to

$$(6) \quad I_T = F^T I_L$$

where I_L and I_T are the vectors of link currents and tree branch currents, respectively. Equation 6 is significant because it isolates the tree branch currents of the general network in terms of the link currents. In the same way, since

$$(7) \quad T\ V_b = 0$$

is always true, a simple partition permits

$$(8) \quad \begin{bmatrix} U & F \end{bmatrix} \begin{bmatrix} V_L \\ V_T \end{bmatrix} = 0$$

or

$$(9) \quad V_L = -F\ V_T$$

where, V_L and V_T are the vectors of link voltages and tree branch voltages, respectively. Equation 9 is significant because it isolates the link voltages of the general network in terms of the tree branch voltages. The Q and T incidence matrices have been eliminated.

Equations 6 and 9 indicate that the F matrix can be used to transform the vector of link currents into tree branch currents and the vector of tree branch voltages into link voltages. These relationships also indicate that F contains one row for each network link and one column for each network tree branch. If the tree of the sample network of Figure 2 is indicated by the bold lines, the appropriate F matrix would appear as in Figure 3. The associated elements are used to caption each row and column. Note that the three link voltages may be "read" horizontally from the F matrix as:

VR1 = -VC1 + E1

VL1 = VC1 - VR2

VJ1 = VR2

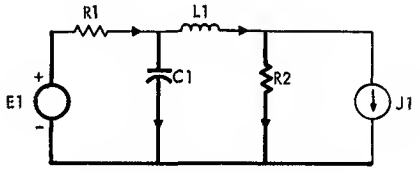


Figure 2. Sample Network

		TREE BRANCHES		
		C1	R2	E1
LINKS	R1	1	0	-1
	L1	-1	1	0
	J1	0	-1	0

Figure 3. F Matrix for Sample Network

These equations are just a restatement of matrix equation 9. Note also that the three tree branch currents may be "read" vertically from the F matrix as:

IC1 = IR1 - IL1

IR2 = IL1 - J1

IE1 = IR1

These equations are a restatement of matrix equation 6. SCEPTRE is programmed to "read" the F matrix that is constructed for each network under analysis and to store the resulting equations[2] for the duration of the problem. These stored equations may then be updated at each solution step. This method may be contrasted with the more conventional approach used in older programs (PREDICT, for example) in which, the same results are obtained by repeated manipulation of the matrix equations at each solution step. When one considers that many problems require literally thousands of solution steps, the computational savings that may be gained by this process are quite significant.

Some of the features of SCEPTRE that represent a significant advance over PREDICT will be found to be unique, or nearly so, if one canvasses all the automatic programs that are available today. It should be realized that this discussion serves only as a summary; a much more complete description exists in [2].

Optional Stored Models

Most contemporary circuits include various types of diodes and transistors that require equivalent circuits with varying degrees of complexity. For example, a large signal equivalent circuit for a transistor usually requires four to six passive elements and at least three to four current generators. If the computer program lacks a stored model feature, the user has no recourse but to insert each component individually into the overall circuit. A partial remedy for this problem is the provision of an "exclusive" stored model of conventional form for diodes and transistors. The word "exclusive" is used in the sense that the user must use whatever topology is originally provided in the stored model; no provision is made to allow topological choice. A far better solution is called the optimal stored model feature. The user may himself, by a simple procedure, store almost any[3] equivalent circuit that is composed of; resistors, capacitors, inductors (including mutual) inductances), and voltage and current sources. Once stored, any model may be called out for use as many times are required in a given circuit application. This feature is not limited to equivalent circuits of active devices; frequently used subnetworks such as biasing networks and filter sections may also be stored.

Rerun Capability

It is often desirable to know how well a given network would function if one or more of its components or forcing functions were varied in size. Separate runs can always be made to determine this, but a better way now exists. A master run is first described, which includes the nominal values of all components and forcing functions. The user appends to this description a list containing the number of reruns desired and the elements that are to be changed for each rerun. No topological changes are permitted between the master run and any of its associated reruns.

Since certain operations are common[4] between the master run and its reruns, machine time economy can be effected by simply performing these operations once, for all of the runs. It should be mentioned, however, that this rerun facility does not imply that anything like true statistical analysis is economically feasible. Solution time per run is still often measured in minutes and therefore, a large number of reruns can become very expensive.

Automatic Termination

In some instances, the user will have no interest in carrying a transient run to completion if it is known that some voltage or current has exceeded, equaled, or dropped below some other network quantity or criterion. A single entry into the program can set up the logic necessary to monitor and automatically terminate any run. Practical applications could be concerned with turn-on, turn-off, or saturation of a transistor or any specific current or voltage limit that the user would care to set. This feature may be used concurrent with the Optional Stored Model and Rerun features.

Flexibility

Some idea of the trend toward flexibility has already been discussed in the section on Optional Stored Models, but there is considerably more to the subject. Sources may be inserted at almost any point in a network and multiple independent variables may be used to control

---

2. The topological limitations that apply to this method are given in [5].

3. Network configurations that include voltage source loops or current source cut sets are prohibited.

4. The tree structure, F matrix, and basic equation setup do not change between runs unless the network topology is changed.

these sources. In addition, a special section has been built into the program to permit the definition of pseudo circuit parameters. To illustrate, the user may determine the power dissipation in resistor R1 of any network by defining the parameter PR1 = (R1 * IR1 ** 2). The indicated mathematical operation would be performed and output at each solution increment. The same feature allows the user to solve systems of first order differential equations that may or may not have any connection with an electrical network. Consider the system

$$\dot{W}1 = 3W1 + 2W2 + 7$$

$$\dot{W}2 = W1 - 3W2 + 3W3$$

$$\dot{W}3 = 10W2 - 20W3 + 5$$

where, $W1(t_0)$, $W2(t_0)$, $W3(t_0)$ are all known. The complete transient response to this system can easily be obtained with just a few input data cards.

Subprogram Capability

It appears to be obvious that no program can be devised to accommodate every operation that all users may require. The next best solution is to write the program so that users can insert subprograms of their own design which will perform special purpose tasks. This option is available to SCEPTRE users with the stipulation that all such subprograms are subject to the rules and requirements of Fortran IV.

The preceding paragraphs by no means exhaust the number of special features that have been incorporated into SCEPTRE, however they should serve to convey some idea of the direction in which automatic circuit analysis is moving. From all indications, development of programs of this type will continue at a rapid rate in the foreseeable future.

References

1. Automated Digital Computer Program for Determining Responses of Electronic Systems to Transient Nuclear Radiation, Vol. II, International Business Machines Corporation, IBM File No. 64-521-5, Owego, New York, July, 1964.

2. Automated Digital Computer Program for Determining Responses of Electronc Systems to Transient Nuclear Radiation, Vol. I, IBM File No. 66-928-611, Owego, New York, September, 1966.

3. S. Seshu and N. Balbanian, Linear Network Analysis, New York: Wiley & Sons, Inc., 1959.

4. T. R. Bashkow, The A Matrix, New Network Description, IRE Transactions on Circuit Theory, Vol. CT-4, September, 1957, pp. 117-120.

5. Automated Digital Computer Program for Determining Responses of Electronic Systems to Transient Nuclear Radiation, Vol. II, IBM File No. 66-928-611I, Owego, New York, Ssptsmber, 1966.

COMPUTER-ASSISTED CIRCUIT ENGINEERING

By

JOHN A. ZUMBADO

Mr. Zumbado is a Project Engineer with International Business Machines, Federal Systems Division, Huntsville, Alabama. He received a Bachelor of Electrical Engineering degree from New York University in 1959.

Since joining IBM in 1963, Mr. Zumbado has been engaged primarily in system and circuit analysis, product improvements, and major development of missile control components.

For three years with Hayes International Corporation, Mr. Zumbado performed design and development tasks on systems and circuits. In previous employment, he performed system analysis and design on a navigation system, and designed electromechanical components for power distribution systems.

JOHN B. EGGERLING

Mr. Eggerling is a Senior Associate Engineer with International Business Machines, Federal Systems Division, Huntsville, Alabama. He received a B. S. degree in Electrical Engineering from the University of Wisconsin in 1962.

From 1962 to 1963 he worked in Product Engineering at IBM SDD, Rochester, Minnesota. From 1963 to the present, he has worked with computer-assisted circuit analysis on Saturn instrument unit control electronics. He recently co-authored a a paper entitled "A Numerical Method for Determining Nodal Voltages in D.C. Circuits with Nonlinear elements."

COMPUTER-ASSISTED CIRCUIT ENGINEERING

By

J. A. Zumbado, Project Engineer
J. B. Eggerling, Senior Associate Engineer

International Business Machines Corporation
Federal Systems Division
Space Systems Center
Huntsville, Alabama

SUMMARY

Accurate and timely circuit engineering is based upon: (a) accurate circuit parameter characterization, (b) flexible network analysis computer programs, (c) continuous computer program development, and (d) empirical investigations in adequate laboratories. This paper shows how these criteria are applied to obtain meaningful circuit analysis results. Typical examples of circuit performance obtained through computer-assisted analyses are given and compared to measured results.

INTRODUCTION

This paper describes the capabilities an engineering organization must consider to perform circuit design and analysis efficiently with the aid of computers. The availability of a large computer system is assumed.

Circuit engineering considers the worst-case circuit performance, which is a function of piece-part tolerances, temperature environment, and aging. Circuit performance criteria include dc stability, power dissipation, and dynamic performance. Dynamic performance encompasses transient behavior, dynamic stability, and four-terminal network characterization of module circuitry. While emphasis will be given to circuit analysis and design, system design and analysis can be and has been performed using the basic approaches described herein.

Accurate and timely circuit engineering is based upon: (a) accurate circuit parameter characterization, (b) flexible network analysis computer programs, (c) continuous computer program development, and (d) empirical investigations in adequate laboratories.

ACCURATE CIRCUIT PARAMETER CHARACTERIZATION

Overall accuracy in the computer programs used emphasizes the need for accurate parameter data of all piece parts, information that is seldom available from device manufacturer's specifications. Models giving accurate device representation can be used because of the increased capability afforded to the engineer by the computer. Device parameters, therefore, must be chosen to fit the models and then measured. For transistors and diodes these parameters are measured on a limited sample (25 devices) at 25 degrees centigrade, and statistical techniques are used to approximate worst-case parameter limits. Data extrapolation for other operating temperatures is accomplished by semi-empirical relationships, developed to reduce parameter measuring costs. Integrated circuit parameters are measured, depending on the module configuration, and are defined as four-terminal network parameters on analog devices, or as switching times and amplitudes for digital devices. In special cases, integrated circuit parameters can be measured by gaining access into the chip; but, after the measurements, the device is no longer usable.

For dc and transient analysis, the nonlinear transistor model (Figure 1) that is based on the Ebers-Moll large signal switching model is used. The model is valid for the three regions of transistor operation: cutoff, active, and saturation. The parameters required, I_E vs V_{BE}, I_{CB} vs V_{BC}, $C_{b'c}$ and $C_{b'e}$, are readily obtained through laboratory measurements.

The hybrid-Pi ac small signal transistor model shown in Figure 2 indicates the circuit configuration, model parameters, and the semi-empirical equations relating temperature dependence. Since the model parameters are interdependent, they must be defined in terms of the basic independent parameters to determine worst-case parameter variations. The basic parameter variations are obtained from the common-base and common emitter h parameters measurements, which are accurate and readily obtained within the laboratory.

NETWORK ANALYSIS COMPUTER PROGRAMS

Experience has shown that circuit analysis programs, to be useful in circuit engineering, must have the capacity to handle large and complex circuits, such as the circuits shown in Figures 3 and 10. Four basic programs are used, depending on the circuit

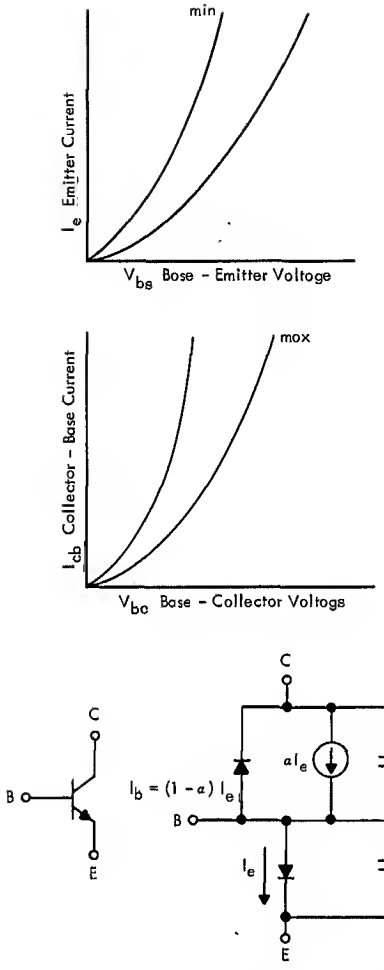


Figure 1.-Nonlinear Transistor Characteristics and Equivalent Circuit, DC and Transient Analyses

performance required. The program characteristics are given in Table 1.

For linear circuits, the Performance Analysis of Electrical Circuits program (PANE) is used to obtain dc stability, bias levels, peak and average power dissipation, and frequency and phase response. The circuit

Table 1--Program Capabilities

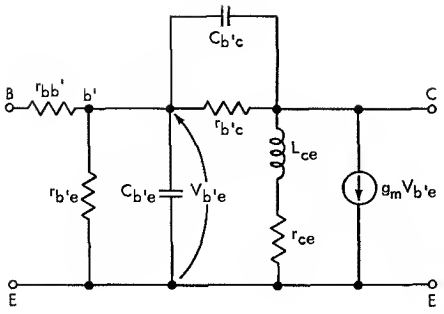
	AC-DC PANE/360	TRANSIENT	CONTROL SYSTEM S-PLANE/360	ANALOG SIMULATOR DSL-90
INPUT PARAMETERS	200	200	--	--
OUTPUTS	50	100	--	50
DEPENDENT NODES	60	50	20	--
DISTRIBUTIONS SHAPES	5	--	--	--
TABLES FOR NONLINEAR INPUTS	20	15	--	--
TRANSFER FUNCTION ORDER	--	--	25	--
FORCING FUNCTION	--	--	5	--
STANDARD FUNCTION GENERATORS	--	--	--	25
SIMULATION BLOCKS	--	--	--	50
MEMORY REQUIREMENTS (BYTES)	256K	220K	200K	150K

*PREDICT/360 incorporates all the problem-solving capabilities of PREDICT I, which was developed by IBM under contract AF29(601)-6309 for the Research and Technology Division, Air Force Systems Command, Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico.

analysis is performed automatically by the program from a topological circuit description, requring a minimum of programming knowledge. Tabular and graphical outputs of worst-case performance or statistical performance are obtained. "Piece-wise" linear dc analysis is performed, using this program, for analysis of switching circuits to determine ON and OFF states, and peak and average power dissipations.

PREDICT/360, a transient analysis program, is capable of computing circuit responses, as a function of time, to any periodic or nonperiodic forcing function which can be defined by a table or equation. The program outputs are time history tabulations or plots or both. The program in its present form is not a worst-case analysis program and engineering judgment has been used to determine circuit worst-case performance. By considering the worst-case parameters that would degrade or improve the circuit and by rerunning the program, the worst-case performance, limited only by the degree of engineering judgment, can be determined.

Linear circuit or control system dynamic response, including root-locus analysis, is obtained by the S-PLANE program. Similar to PREDICT, S-PLANE obtains the transient response of a transfer function to an impulse, sine wave, square wave, ramp, or exponential input. The transfer function of a network can also be obtained.



$$*r_{bb'} = r_b \qquad K_1 = \frac{\frac{h_{rb}}{h_{ob}} - r_b}{(1+h_{feo})(r_e)}$$

$$r_{b'e} = h_{feo} \cdot r_e \qquad r_{ce} = \frac{r_e}{h_{feo} \cdot K_1}$$

$$C_{b'e} = \frac{1}{r_e \cdot \omega_\alpha} \qquad L_{ce} = \frac{r_c}{\omega_\alpha \cdot K_1}$$

$$*C_{b'c} = C_{tc} \qquad r_{b'c} = \frac{r_c}{1 - K_1}$$

$$g_m = \frac{h_{feo}}{(1+h_{feo})(r_e)} \qquad r_e = \frac{\frac{h_{ie}}{h_{fe}} - r_b}{h_{fe}+1}$$

$$r_c = \frac{1}{h_{ob}}$$

*Meosured parameters

Semi-empirical Equations Showing Temperature Dependence
$r_b(T) = r_{b25°}\left(\frac{T+273}{298}\right)^{2.6}$
$h_{feo}(T) = h_{feo25°}\left[1+.007(T-25)\right]$
$r_e(T) = r_{e25°}\left(\frac{T+273}{298}\right)$
$C_{tc}(T) = C_{tc25°}\left(\frac{T+273}{298}\right)^{2.25}$
$r_c(T) = r_{c25°}\left[1+.0027(T-25)\right]$
$\omega_\alpha(T) = \omega_{\alpha\ 25°}\left(\frac{T+273}{298}\right)^{-1.4}$
$K_1(T) = K_{1\,25°}\left(\frac{T+273}{298}\right)^{-1.2}$

Note: "T" is in degrees centigrode.

Figure 2.-*Hybrid-Pi Equivalent Circuit

*Bymers R. J. "Linear Amplifier Design Manuol," M05-0011-0, IBM, Standords Engineering, Generol Products Division, Glendale Laborotories

The dc amplifier (Figure 3) is an analysis example demonstrating the performance characteristics determined, using computer-aided circuit analysis techniques. The analysis determined the amplifier open-loop performance characteristics and, from these characteristics, a mathematical model was developed. Ths model was then used to determine the amplifier closed-loop performance. The worst-case dc null offset was determined for the closed-loop amplifier as a function of temperature. Figures 4 and 5 sbow the computed amplifier open-loop voltage gain magnitude and phase along with laboratory measurements for comparison of computed and measured results. The amplifier worst-case open-loop voltage gain magnitude response is shown in Figure 6. Results of the worst-case analysis of the amplifier indicated a marginal design with respect to amplifier stability. The amplifier was then stabilized with additional compensation. The characteristics of the improved amplifier are shown in Figure 6, depicting a worst-case phase margin of 40 degrees. Figure 6 also sbows the results of the dc null offset analysis and the open-loop-mathematical model derived. The model performance characteristic is also plotted for comparison. The clossd-loop performance of the amplifier is shown in Figures 7, 8, and 9. The circuit configuration analyzed is shown in each figurs, as is the minimum, nominal, and maximum circuit response.

For circuits or systems having highly nonlinear elements (such as level detectors in combination with amplifiers), the circuit is broken into individual blocks that can be analyzed individually by the programs mentioned. It is this division that requires a high degree of engineering skill, judgment, and experience on the part of tbe analyst. These blocks are then charaoterized as four-terminal networks including all nonlinearities, thus allowing a system simulation to bs obtained by using the analog computer simulation program DSL-90.

COMPUTER PROGRAM DEVELOPMENT

While some of the programs stress that the lack of programming knowledge does not impede their use, experience has shown that being able to modify the programs to fit a particular circuit under consideration has produced timely and quite unexpected results. Because of the increased flexibility afforded by the programs, the engineer can evaluate his design or complete his analysis, without ignoring important but often neglected parameters, if he can fit the program to the circuit under scrutiny. This program knowledge and the use of these programs have resulted in a continued program updating to enable the analysis programs to keep abreast of circuit technology. In addition, this knowledge has resulted in continued work to reduce computing time and to increase the flexibility of the programs (i.e., adapting steepest descent techniques to PREDICT/360 now under consideration, to enable worst-case transient

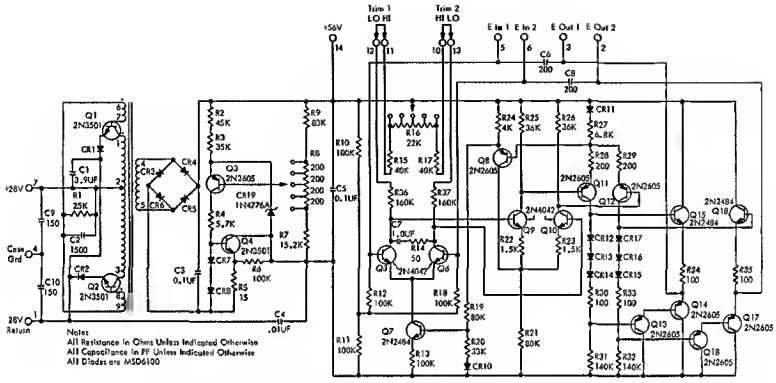


Figure 3.-Schematic of Complete DC Amplifier

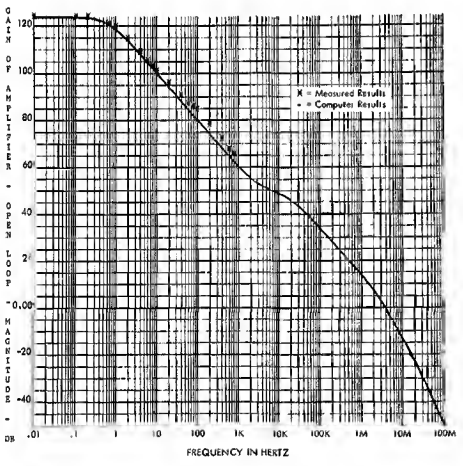


Figure 4.-Gain of Open-Loop Amplifier Magnitude (-55°C)

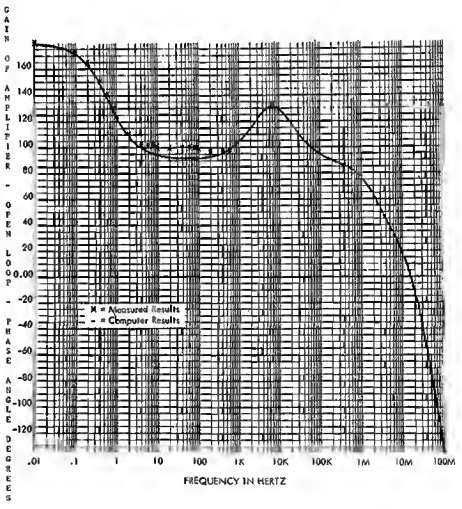


Figure 5.-Gain of Open-Loop Amplifier Phase Angle (-55°C)

analysis). Program development over the past four years has increased from the analysis capability of circuits having a maximum of 11 dependent nodes on a statistical basis to the capability afforded by the PANE program. Along with the ac and dc capability of PANE, other programs were introduced and updated to their present capabilities as shown in Table I.

EMPIRICAL INVESTIGATIONS

Often, especially during design efforts when timely analysis results are required, empirical circuit responses must be obtained, since mathematical models

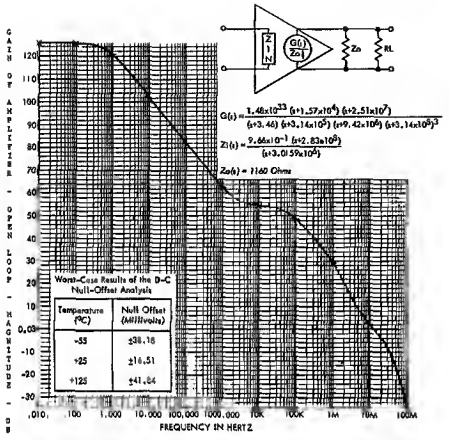


Figure 6.-Gain of Open-Loop Compensated Amplifier (-55°C)

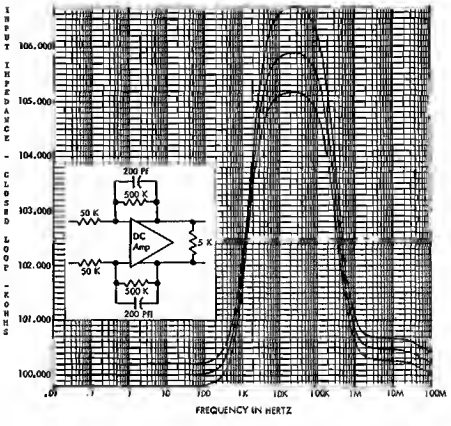


Figure 8.-Input Impedance of Closed-Loop Amplifier Magnitude (C = 0, +125°C)

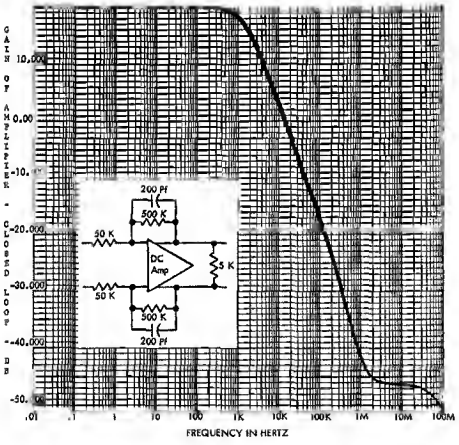


Figure 7.-Gain of Closed-Loop Amplifier (C = 0, +125°C)

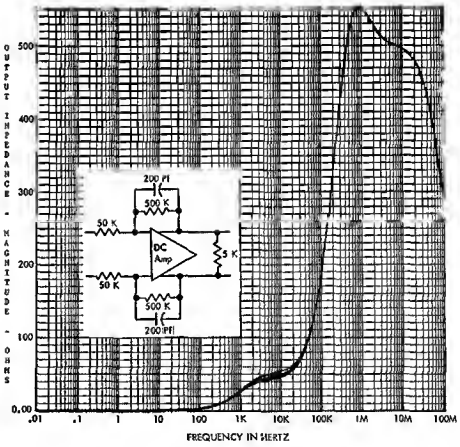


Figure 9.-Output Impedance of Closed-Loop Amplifier Magnitude (C = 0, +125°C)

cannot be developed in the time frame of the analysis. An adequately equipped electronic measuring laboratory, therefore, becomes an integral part of computer-assisted circuit engineering.

A servo amplifier system having nonlinear elements is shown in Figure 10. The amplifier was divided into individual blocks, as shown in Figure 11. The magnetic amplifier and load, being nonlinear blocks, were

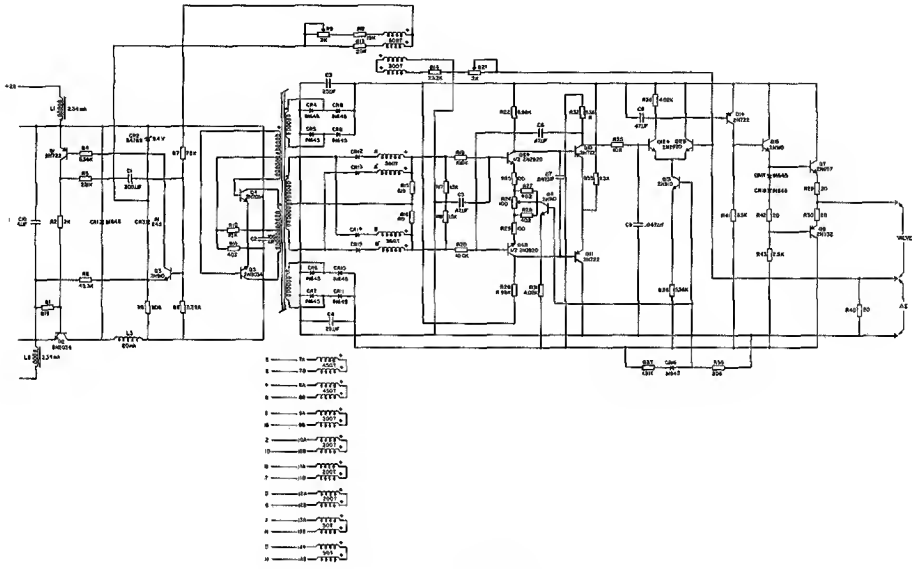


Figure 10.-Electrical Schematic of 50 MA Servo Amplifier

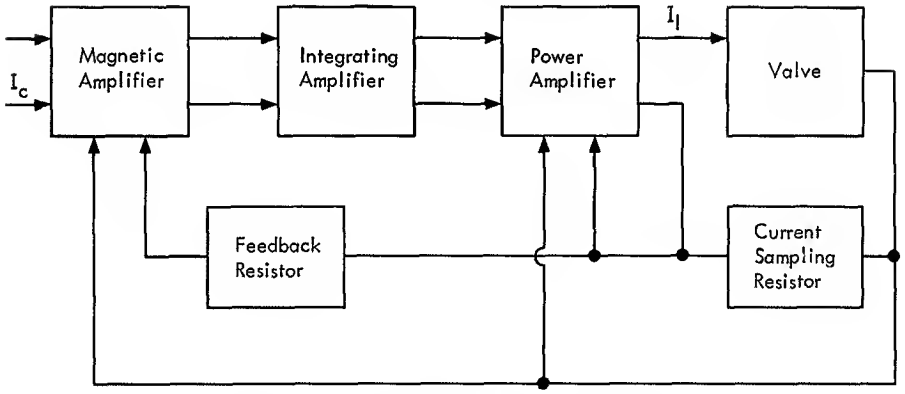


Figure 11.-Descriptive Block Diagram of 50 MA Servo Amplifier

characterized using empirical techniques. The empirically determined load characteristic, driving point impedance function, and synthesized circuit used for the analysis are shown in Figure 12. The magnetic amplifier input impedance is shown in Figure 13. After all blocks were characterized, the amplifier was

analyzed for closed-loop performance. From the closed-loop performance characteristics, a mathematical model for the amplifier was derived as shown in Figure 14. The worst-case current gain performance characteristics are shown. The broken line depicts the minimum gain at the temperature design limit of the amplifier. Laboratory measurements are plotted to show correlation between computed and measured results.

CONCLUSIONS

Computer-assisted circuit engineering, therefore, requires more than just one computer circuit analysis program. The circuit parameters must be known. Several programs must be available and checked out in the computer laboratory. Program development must keep the programs compatible with circuit technology. Finally, a laboratory for parameter measurement, empirical evaluation of circuit blocks, and analysis reasonableness checks must be an integral part of computer circuit engineering.

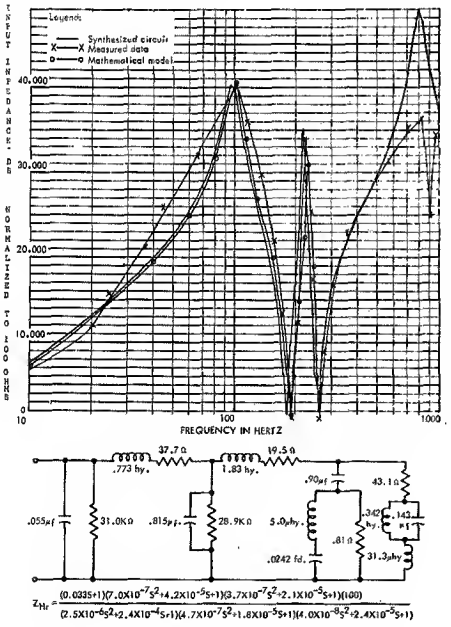


Figure 12.-50 MA Servo Amplifier Load

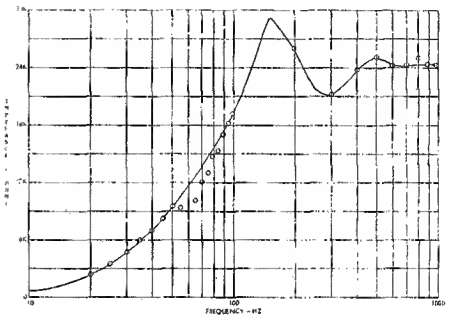


Figure 13.-Magnitude of Input Impedance of 450 Turn Control Windings

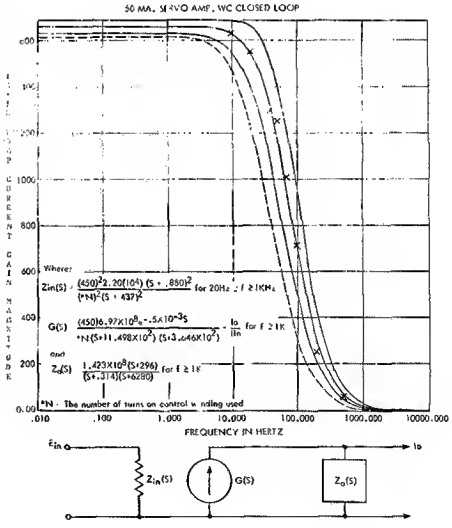


Figure 14.-50 MA Servo Amplifier WC Closed-Loop Performance

OPTIMIZATION WITH COMPUTERS

By

WILLIAM G. SCHEERER

Mr. Scheerer received a B. E. E. degree in 1959 from Syracuse University, and a M. S. E. E. degree in 1960 from California Institute of Technology.

He was with the Bell Telephone Laboratories from 1960-1961; U. S. Army from 1961-1963; and the Bell Telephone Laboratories from 1963 to present.

Mr. Scheerer was first concerned with wideband, transistorized amplifiers, and then with systems studies for medium- and high-speed PCM systems. Since 1964 Mr. Scheerer has led a group concerned with device modeling, application of digital computers to engineering problems, and network development.

He is a member of the IEEE, ACM, Sigma Xi, Tau Beta Pi, and other honorary societies.

OPTIMIZATION WITH COMPUTERS

By William G. Scheerer

Bell Telephone Laboratories, Inc.
North Andover, Massachusetts

SUMMARY

The widespread availability of high-speed digital computers has profoundly affected many technical areas, and the impending introduction of powerful time-sharing systems will again revolutionize the engineer's work. One field of considerable importance is the area of optimization or approximation; for example, design of loss, phase, and delay equalizers is usually accomplished by iterative techniques. Many optimization algorithms have been designed and tested. Each possesses particular advantages and disadvantages. Search techniques such as grid, random, and pattern search and the simplex algorithm are less affected by certain function anomalies than slope-following methods such as steepest descent, gradient partan, Fletcher-Powell and generalized Newton-Raphson. Steepest descent and gradient partan usually work with a poor initial estimate of the solution, while the generalized Newton-Raphson approach is efficient near the optimum, but may diverge elsewhere. The Fletcher-Powell technique utilizes both first and second derivative information, and is quite often the most efficient. Not all of these methods allow ready inclusion of constraints. Ideally the designer should have available a variety of algorithms, and should be able to apply any combination to his problems.

INTRODUCTION

The digital computer makes practical iterative design and optimization procedures where only an engineer's judgment could have been used previously. Although overall system optimization is not yet achievable, subsystem optimization is used every day in practical engineering problems. The introduction of direct-acceaa time-sharing computer systems will again increase the range of problems wherein iterative optimization techniquea are useful.

In approaching an iterative approximation or optimization problem the engineer has many techniques from which to choose, varying from some brute force search techniques to highly sophisticated second order gradient algorithms. Each approach has its advantages and disadvantagea. Several techniques will be briefly described in this paper, followed by a description of an existing program with sample results.

For more detailed descriptions snd comparisons of optimization algorithms the reader ia referred to a number of excellent articles[1-6] and books[7-9]. Reference 8 presents results of recent practical work, comparisons of methods, and an unusually complete bibliogrsphy. The contents of this paper hsve been presented in expanded form.[10]

THE OPTIMIZATION PROBLEM

The function to be optimized is of the form

$$y = y(f_1,\ldots,f_\ell, x_1,\ldots,x_n) \qquad (1)$$

where $\underline{f}$ is the set of functional parameters and $\underline{x}$ is the set of parameters which may be varied to optimize the function y. Usually y is to be maximized or minimized; in this paper it is assumed that y is to be minimized.

In general y can be any function of $\underline{f}$ and $\underline{x}$ that is to be minimized. In practical optimization problems y usually takes the form of an error measure which is a complicated function of $\underline{f}$ and $\underline{x}$. One popular functional form is the so-called "least squares" error criterion

$$y = \sum_i w_i(g_i - r_i)^2 = \sum_i w_i e_i^2 \qquad (2)$$

where

y = error measure to be minimized,

e_i = unweighted error at f_i,

$g_i = g(f_i, x_1, \ldots, x_n)$ = computed values,

x_j = j th parameter of the n parameters to be varied,

$r_i = r(f_i)$ = required or desired function value,

f_i = i th value of independent variable, and

w_i = assumed weighting function (allows placing emphasis on particular requirements).

Another useful criterion is the maximum absolute error (Chebyshev error):

$$y = \max_i |w_i(g_i - r_i)| \quad . \tag{3}$$

In both cases the function y is often a complicated expression or group of expressions. Implicit in both error criteria is the existence of requirement data $\underline{r}$ (e.g., loss equalizer requirements) to which computed values $\underline{g}$ (e.g., loss of equalizer configuration) are to be matched by varying parameters $\underline{x}$ (e.g., element values of loss equalizer).

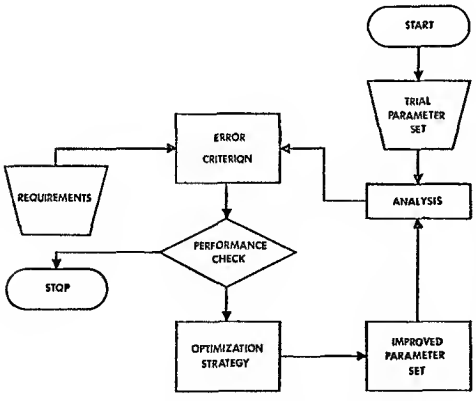


Figure 1.-Optimization Program Block Diagram

Figure 1 outlines the steps required to solve typical engineering problems iteratively, and hence is also an outline for a computer program to automate the task.

OPTIMIZATION METHODS

The two general classes of optimization algorithms are the slope-following and search techniques. The slope-following methods use a gradient or similar slope-determining function to find a new set of parameters which will reduce y. Search techniques are varied, but do not use gradient information.

Slope-following techniques are often superior for functions with continuous derivatives, but are more time-consuming because of the large number of derivatives to be found. The search techniques, however, often work better with functions with discontinuous derivatives. In addition, constraints are often easier to include and, in most cases, a greater variety of optimization criteria can be incorporated. All optimization schemes except some random and grid search methods tend to reach the nearest local minimum of the function to be optimized.

In the following sections the unrealistically simple function

$$y = 16\, x_1^2 + (x_2 - 4)^2 \tag{4}$$

will be used to illustrate various optimization techniques.

SEARCH METHODS

Grid and Random Search

Grid search basically involves evaluating Equation (1) for many choices of $\underline{x}$ with each x_i taking on values throughout a prechosen range, as illustrated on Figure 2. In random search the trial parameter values are chosen with some degree of randomness; in the illustration on Figure 3 the direction and distance of each move are chosen at random, and only those moves resulting in a lower y are retained.

Pattern Search

Pattern search techniques attempt to establish proper moves (succession of changes of $\underline{x}$) by first making a series of exploratory maneuvers. After exploring the local terrain a larger move called a

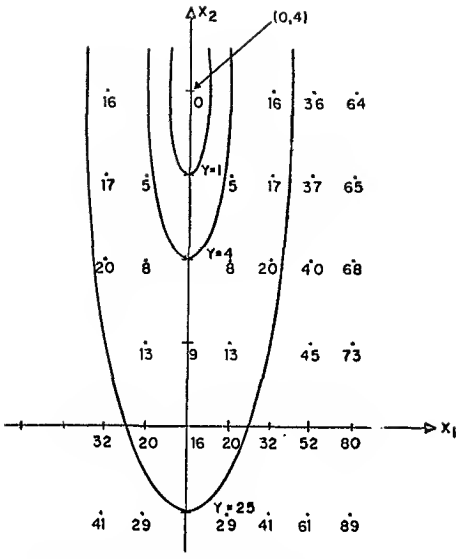


Figure 2.-Grid Search Method

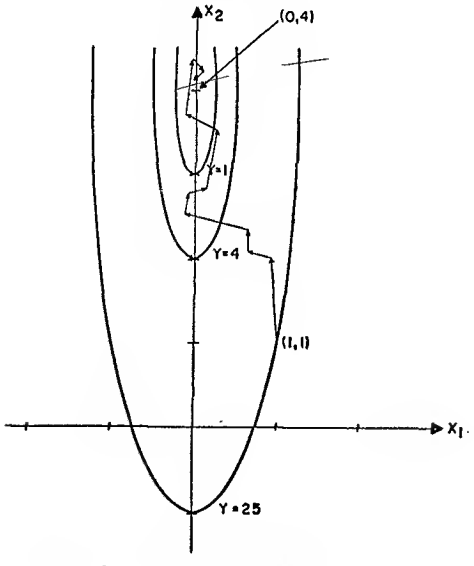


Figure 3.-Random Search Method

pattern move is made based on the results of previous trials. Many methods of accomplishing the local exploration and choosing the accelerating moves have been tried. A test result for one of the approaches (see Ref. 9, pp. 146-150) is illustrated in Fig. 4.

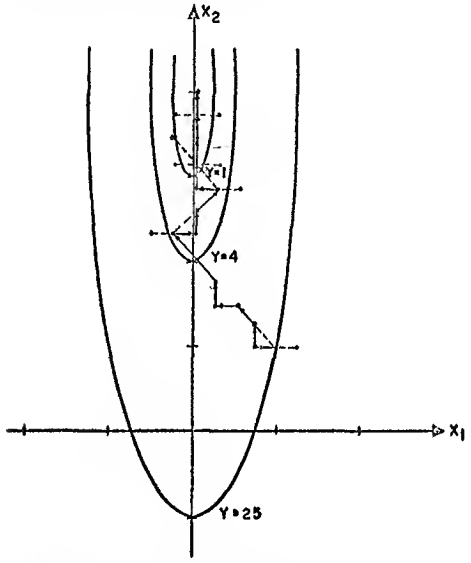


Figure 4.-Pattern Search Method

Simplex Method

The Simplex algorithm[12] is outlined on Figure 5. Unlike the other techniques which involve successive sets of $\underline{x}$, the Simplex method always works with (n+1) sets of $\underline{x}$, where n is the number of parameters to be varied. The object is to continually replace the set $\underline{x}_w$ producing the worst (highest) y with a new set through four basic actions: reflection (new $\underline{x}_r$ chosen by reflecting $\underline{x}_w$ about the center of gravity $\underline{x}_o$ of all $\underline{x}$ except $\underline{x}_w$), expansion (new $\underline{x}_e$ chosen further in direction of previous successful reflection), contraction (intermediate $\underline{x}_c$ chosen following unsuccessful reflection), and shrinking (moving all $\underline{x}$ closer to the set leading to minimum y when first three actions fail). The sample problem leads to the initial simplex iterations shown in Figures 6 and 7.

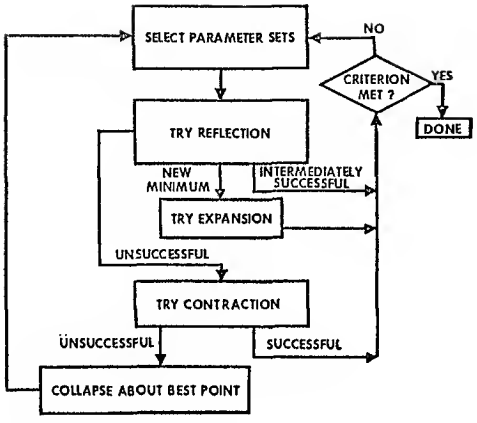


Figure 5.-Simplex Method-Basic Algorithm

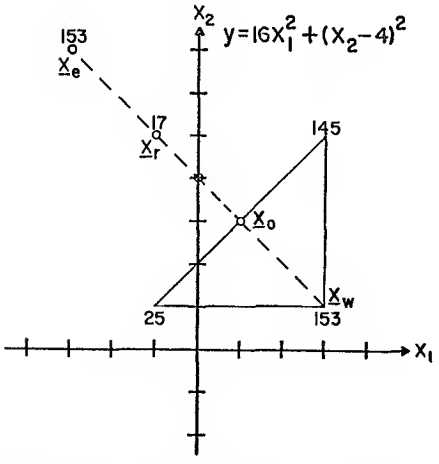


Figure 6.-Simplex Example - First Step

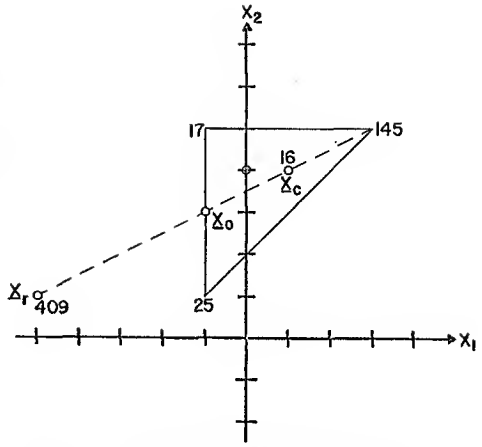


Figure 7.-Simplex Example-Second Step

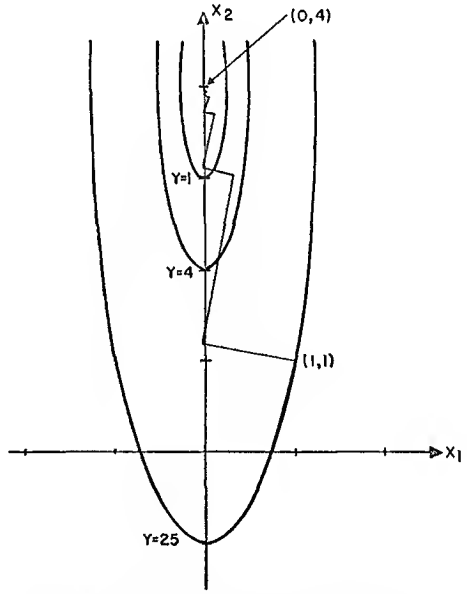


Figure 8.-Unmodified Steepest Descent Method

SLOPE-FOLLOWING METHODS

Steepest Descent Method

The steepest descent method relies on the elementary calculus notion that the gradient of a function points in the direction of increasing functional values. Hence to minimize a function successive parameter sets can be calculated as

$$\underline{x}_{i+1} = \underline{x}_i - k\, \underline{\nabla y}(\underline{x}_i) \ . \qquad (5)$$

Figure 8 illustrates a steepest descent path. The gradient partial derivatives also are used in many other algorithms, including the gradient partan, generalized Newton-Raphson, and Fletcher-Powell methods.

Gradient Partan

Gradient partan[13,14] is one of several versions of the method of parallel tangents. The method seeks to speed up the steepest descent approach by combining the results of gradient steps to choose a direction leading to a large improvement, as shown for the test problem on Figure 9. Considerable acceleration is often experienced.

Generalized Newton-Raphson Techniques

When $\underline{x}$ is close to the optimum most functions $\underline{y}$ are approximately quadratic. In the generalized Newton-Raphson technique each x_i is replaced by $x_i + a_i \Delta x_i$. The partial derivative of the least squares error measure with respect to each a_i is then calculated and set equal to zero. The resulting equations are linear in $\underline{a}$, and can be solved. If the function g is nonlinear in $\underline{x}$, as is usually the case, this procedure must be iterated.

Fletcher-Powell Algorithm

The Fletcher-Powell algorithm[15] introduces a matrix H which is usually initially chosen to be a unit matrix. At each iteration $\underline{H}$ is varied, and it approaches the inverse of the matrix of second derivatives. Instead of searching in the negative gradient direction for a minimum, a search is made along

$$\underline{S} = -(H)(\underline{\nabla y}) \ . \qquad (6)$$

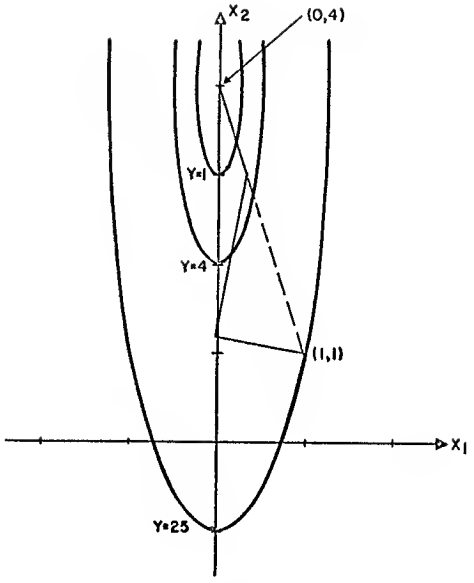


Figure 9.-Gradient Partan Method

A COMBINATION OPTIMIZATION PROGRAM

The SUPROX Program

The SUPROX (SUccessive apPROXimation) program[16] combines the modified steepest descent algorithm with the generalized Newton-Raphson technique in a general program package.* The user must supply a routine to evaluate the function g, unless he can use one of the many routines in the program library file. Many specified options are included, including limiting parameter variations to specified ranges, weighting requirements, and using requirement ranges. The program has been effective on many practical engineering problems, although no program has yet been found which will solve all problems.

*The organization of the SUPROX program is due to Mr. C. L. Semmelman of Bell Telephone Laboratories, Holmdel, New Jersey.

Sample Problem - Delay Equalizer Design

A delay equalizer design problem is used to illustrate the use of SUPROX. An eight-section delay equalizer was required as shown in Figure 10. The SUPROX function evaluation subroutine works with ideal all-pass sections, and a subsequent program computes parasitic-corrected network element values from the idealized parameters.

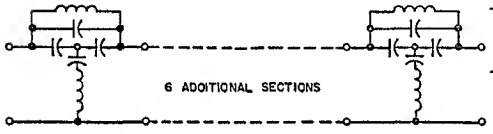


Figure 10.-Eight-section Delay Equalizer

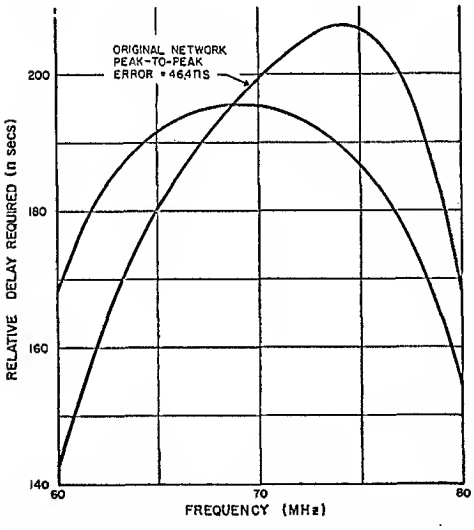
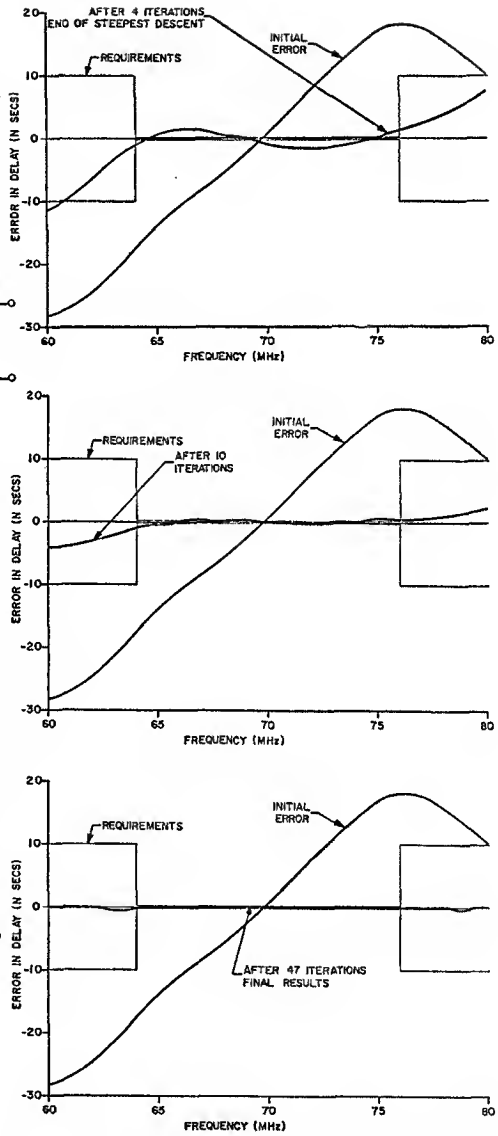


Figure 11.-Delay Equalizer Requirements and Performance of Initial Design

Without any attempt to design the equalizer, initial parameter values were chosen, producing the large deviation between required delay and network performance shown on Figure 11.



Figures 12A,B,C.-Delay Errors at Various Stages of Optimization Process

The error requirements and the original error are shown in Figure 12A, along with the results after four iterations when the modified steepest descent method has ceased producing significant improvements. The program then switched to the generalized Newton-Raphson algorithm. Figures 12B and 12C illustrate the delay error after 10 and 47 iterations respectively. Tha requirements ($\pm$0.2 ns in the center region) were met with greater than a 2:1 margin remaining for physical realization errors.

The performance of SUPROX can also be visualized by examining the match error y and the number of zero crossings of the delay error as a function of the number of iterations, as shown in Figure 13.

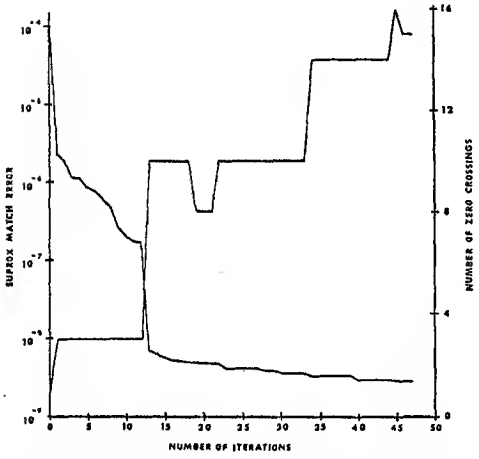


Figure 13.-Match Error and Number of Zero Crossings for Delay Example

CONCLUSION

The optimization techniques discussed in this paper have been applied to practical engineering problems, but none is the desired panacea. The engineer should have access to programs with a variety of methods, and ideally should be able to interact with the computer during the optimization procedure. As new techniques and powerful time-sharing computer systems evolve the optimization procedure may again be dramatically altered.

REFERENCES

1. Box, M. J.: A Comparison of Several Currant Optimization Methods, and the Use of Transformations in Constrained Problems. Computer Journal, Vol. 9, No. 1, May 1966, pp. 67-77.

2. Dickinson, A. W.: Nonlinear Optimization: Some Procedures and Examples. Procaedings of the 19th National Conference, Association for Computing Machinery, 1964, pp. E1.2-1 to E1.2-8.

3. Gilbert, E. G.: A Selected Bibliography on Parameter Optimization Methods Suitable for Hybrid Computation. Seminar on Hybrid Computation, University of Wisconsin, March 17-18, 1966.

4. Spang, H. A., III: A Review of Minimization Techniques for Nonlinear Functions. SIAM Review, Vol. 4, 1962, pp. 343-365.

5. Wood, C. F.: Review of Design Optimization Techniques. IEEE Transactions on Systems Science and Cybernetics, Vol. SSC-1, No. 1, November 1965, pp. 14-20.

6. Zoutendijk, G.: Nonlinear Programming: A Numerical Survey. SIAM Journal on Control, Vol. 4, No. 1, 1966.

7. Balakrishman, A. J., and L.W. Neustadt, editors: Computing Methods in Optimization Problems, New York, Academic Press, 1964.

8. Lavi, A. and T. P. Vogl, editors: Recent Advances in Optimization Techniques. New York, John Wiley & Sons, Inc., 1966.

9. Wilde, D. J., Optimum Seeking Methods, Englewood Cliffs, N. J., Prentice Hall, Inc., 1964.

10. Scheerer, W. G.: Optimization with Computers. Circuit Design by Computer Tutorial Symposium, New York University, January 31 - February 2, 1967.

11. Hook, R., and T. A. Jeeves: "Direct Search" Solution of Numerical and Statistical Problems. Journal

Association for Computing Machinery, Vol. 8, No. 2, April 1961, pp. 212-229.

12. Nelder, J. A., and R. Mead: A Simplex Method for Function Minimization. Computer Journal, Vol. 7, No. 4, January 1965, pp. 308-313.

13. Powell, M. J. D.: An Iterative Method for Finding Stationary Values of a Function of Several Variables. Computer Journal, Vol. 5, No. 2, July 1962, pp. 147-151.

14. Shah, V. B., R. J. Buehler, and O.Kempthorne: The Method of Parallel Tangents (PARTAN) for Finding an Optimum. Technical Report No. 2, Iowa State University Statiatical Laboratory, Ames, Iowa, April 1961 (reviaed August 1962).

15. Fletcher, R., and M. J. D. Powell: A Rapidly Convergent Descent Method for Minimization. Computer Journal, Vol. 6, No. 4, July 1965,pp 163-168.

16. Fleischer, P. E.: Optimization Techniquea. Chapter 6 in Kuo, F. F. and J. F. Kaiser, editors: System Analysis by Digital Computer. New York, John Wiley and Sons, Inc., pp. 175-217.

NASAP: NETWORK ANALYSIS FOR SYSTEMS APPLICATIONS PROGRAM: PRESENT CAPABILITIES OF A MAINTAINED PROGRAM

By

RICHARD M. CARPENTER

Mr. Carpenter is a member of the Design Criteria Branch, NASA Electronic Research Center, Cambridge, Massachusetts. He received his B. S. E. E. degree (cum laude) from Tufts University and his M. S. degree in Applied Mathematics from Harvard University.

Mr. Carpenter's experience with NASA includes circuit design on the Nimbus meteorological satellite at NASA/Goddard Space Flight Center prior to joining the Electronic Research Center. His current assignments include applied research in computer-aided design of standard reliable circuits for use in NASA space missions.

Mr. Carpenter is a member of Tau Beta Pi and Eta Kappa Nu.

NASAP: A NETWORK ANALYSIS FOR SYSTEMS APPLICATIONS PROGRAM: A PRESENT CAPABILITIES OF A MAINTAINED PROGRAM

By
Richard M. Carpenter
NASA Electronics Research Center
Cambridge, Massachusetts

ABSTRACT

A progress report is presented on the first year of a cooperative effort to develop a modularly arranged circuit analysis program by

(a) A group of seven universities with small to medium size computer facilities, each participant specializing in a module of the program.
(b) Several industrial users of computer-programs with extensive experience, nearly all of whom are active in developing their own in-house programs and who are willing first to trade know-how and secondly are willing to give serious thought to cooperative programming efforts.

Present capabilities of NASAP are discussed including flowgraph construction and evaluation, input-output formulation and available subroutines. Strengths and weaknesses are presented, together with plans for future development.

NASAP DEVELOPMENT

Introduction

The Network Analysis for Systems Applications Program (NASAP) has been developed by the NASA/Electronics Research Center as a two-fold project:

(a) A research tool for the development of new concepts in automated circuit design.
(b) An effective means of establishing qualification and standardization procedures for future NASA computer programs.

The first phase of the project has involved the definition and implementation of the symbolic oriented techniques, and an outline of their effectiveness in a modularly arranged program. This work is well documented in the literature (1-12). This portion of NASAP will be discussed in detail and future work will be outlined.

Modular Structure

The NASAP program has been written in FORTRAN IV for the CDC 3600 computer. It has been designed in a modular arrangement, with the main part of the program designed to develop and evaluate a symbolic and numerical frequency-dependent signal flowgraph for a given network. The various modules perform specific applications and extensions of the reduced flowgraph. This arrangement has provided an effective method of dividing the program development among several universities and companies.

The modular structure has also provided for expansion of the program to include additional subroutines or larger systems as the need arises. A more efficient use of computer time is achieved since the engineer selects which of several analyses or outputs he desires for a specific application, and bypasses those which he does not desire.

The program has developed with seven major branches emanating from the flowgraph construction and evaluation program. These correspond to the areas of seven university grants in progress during FY67. Included are various display routines for the derived flowgraph, the addition of subroutines for non-linear elements Bode plot, transient response, sensitivity analysis, tolerance analysis and derivation and construction of an approximate flowgraph. In addition, each university is responsible for qualifying the main portion of the program as well as its own subroutine on its own computer. This results in a program which is qualified on a number of different computers.

Maintenance Procedures

To avoid costly duplication of effort for a large program (on failure diagnostics and extension of scope) an atmosphere for a free exchange of program information is a prime requisite. For NASAP, program information and documentation is made available on the understanding that recipients will freely interchange programs, diagnostics, and additions. NASAP is a cooperative effort of about 20 users, whose development has fostered:

(a) A modular structure of compatible subroutines, such that each subroutine can be improved, extended and elim-inted separately as needed. The maintenance of each module is the assigned responsibility of one or more key participants with particular interests in these subroutines.

(b) Qualification procedures, such as a sequence of sample problems with corresponding performance criteria, such as running times and round-off error for two or more machines. It is also desirable to include a group of problems which reveal at what point the performance of the program becomes marginal or unsatisfactory.

Some preliminary results of this cooperative effort entailed in NASAP will now be discussed.

NASAP PROBLEM FORMULATION

From a given network, NASAP constructs a corresponding block-diagram, often referred to as flowgraph which relates voltages and currents of the network elements. The block-diagram may have frequency dependent or time-dependent, linear and nonlinear, active and passive elements. Active elements are modeled in the block-diagram either as voltage or as current sources which are controlled by the output of other sources.

A passive element is modeled as a special case of an active element, in which source and control are of opposite type; namely, a voltage controlled current source (Y) or a current controlled voltage source (Z). Thus, for each passive element a choice or dichotomy is necessary in modeling the network. This choice is constrained by requiring that no voltage sources are in parallel and no current sources in series.

The flowgraph approach, often referred to as the "dichotomous approach," differs significantly from matrix oriented techniques. Like matrix techniques it serves to determine driving point and transfer functions, but it also provides valuable insight into problem formulation. An added feature is that network functions may be obtained in symbolic form.

The NASAP program constructs a unique flowgraph for a given equivalent circuit. A network element in a flowgraph is represented as a transmittance G (S) relating two variables X and Y as in Figure 1. Evaluation of a transfer or

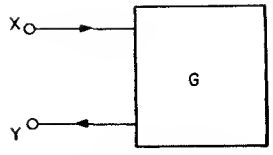


FIG. 1. Open System Representation, Y = GX

driving point function is reduced to the determination of the transmittance between an input and output node of the resultant flowgraph. For instance, for Figure 3 the problem is to determine the voltage gain. It is therefore necessary to determine the transmittance between the input node of the flowgraph (the voltage across element 1) and the output node (the voltage across element 9). In flowgraph notation this is as shown in Figure 1 where $X = V_i$ and $Y = V_o$ for Figure 3. For an actual circuit the corresponding flowgraph will be a many noded construction for which G(S) will not be at all obvious. The computer program evaluates this flowgraph by grouping algorithms which permit symbol manipulation within the computer.

The flowgraph of Figure 1 is open, i.e. starting at the input node (X) there is no way of returning to that node in the direction of the transmittances of the flowgraph. Desired is a flowgraph which has only interdependent variables, such as Figure 2. To close the flowgraph a dummy

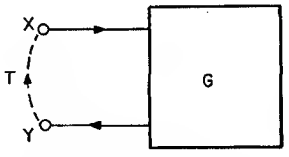


FIG. 2. Closed System Representation, Y = GX, X = TY

transmittance T is used. The program evaluates symbolically a closed system containing the unknown parameter

$$T(S) = X(S)/Y(S) = 1/G(S). \qquad (1)$$

It is instructive to interpret this result in terms of the circuit of Figure 3. To determine

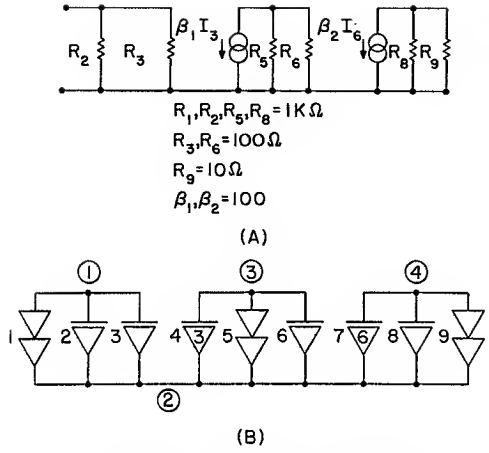


FIG. 3. Example 1: Two-Transistor Amplifier

the voltage gain, the voltage of the input generator, element 1, is made dependent upon the voltage across element 9, which is the output voltage. Hence, the desired transfer function V_o/V_i can be calculated as 1/T(S).

From the constructed block-diagram linear, piece-wise, stochastic, or non-linear routines can evaluate desired circuit criteria. Significant advances result from

(a) Labyrinth and grouping algorithms, which circumvent combinatorial schemes used for tree enumeration in passive networks.
(b) Tagging techniques, which utilize properties of closed systems. For example, in symbolic calculations no distinction is made between known and unknown quantities.
(c) Optimization procedures, which eliminate from a circuit those design parameters which do not affect performance parameters more than a preassigned level of accuracy. Thus, the simplest possible model is displayed, given a set of design requirements.

Network and transfer functions as well as sensitivity and generating functions are thus obtained symbolically and numerically in the frequency domain. State-space techniques yield the corresponding transient response in the time-domain or in the probability domain.

Various display techniques can be used to construct Bode plot, pole-zero configuration and similar design graphs.

NASAP INPUT-OUTPUT AND EVALUATION ROUTINES

The tutorial description of NASAP to follow will outline the fundamental algorithms and describe in general terms the mechanics of the program, viz, :

Coding of equivalent circuit (input statement)
Construction and evaluation of flowgraph
Function and operation of available modular subroutines
Illustration by example

The Equivalent Circuit

The first step in preparing a circuit for computer analysis is the construction of a coded equivalent circuit from the given schematic. This serves one main purpose: the separation of all elements into two distinct groups (current generators coded "1", and voltage generators coded "0"). For passive elements, admittances and impedances are considered as voltage controlled current generators and current controlled voltage generators respectively. This dichotomy or separation of elements into two mutually exclusive groups is essential to the flowgraph construction by the computer.

The elements of the coded schematic are numbered consecutively as are the independent voltage nodes. A group of elements selected as voltage generators is called the "tree"; those selected as current generators the "co-tree". A separation of the network elements into these two mutually exclusive groups, tree and co-tree, is made subject to certain constraints imposed by network physics:

(1) Voltage sources in the circuit are encoded as voltage generators. Thus in the coded equivalent circuit, voltage sources always form part of the tree. Conversely circuit current sources are encoded as current generators in the coded equivalent and hence never form part of the tree.
(2) A passive element if contained in the tree is referred to as an impedance, (Z). If contained in the co-tree it is referred to as an admittance, (Y).

(3) When assigning elements to the tree and co-tree, voltage sources are never allowed in parallel or current sources in series.

Coding of Equivalent Circuits for Computer Input

The dichotomized equivalent circuit must be presented as input data to the program in the form of a matrix specifying circuit topology, frequency dependence, numerical values, and dichotomous structure. Table 1 gives this problem statement for the circuit in Figure 3A. Each row corresponds to one circuit element numbered as in Figure 3B, and indicated in input E.

Column A describes the vertex of origin of E, where the direction of the arrows in Figure 3B specifies the assumed positive direction of current. Column B specifies the vertex termination. Thus element 1 has entry A = 1, B = 2 to specify the positive direction of current as flowing from node 1 to node 2. The inputs A and B thus describe the circuit topology.

Column C indicates the dichotomy of voltage or current control to the program by the binary inputs "0" for voltage and "1" for current. Input D indicates the element which is the controller. This is the same as E for a passive element since a passive element performs its own control function between input and output. For the active controlled source E = 4, the current is controlled by element 3, hence D = 3. Input G serves to complete the dichotomous description in the same binary manner as C. For example, from Table 1 and Figure 3B element 3 is a passive voltage controlled current generator, a resistor, hence is coded C = 0, D = 3, G = 1.

As discussed in the second section (NASAP Problem Formulation), a closed flowgraph for the circuit is desired. This is specified in input H which identifies the unknown parameter, and indicates how the system is to be closed by the dummy transmittance T (S). For the example, the voltage gain is desired. Referring to (1), $X(S) = V_i$, $Y(S) = V_o$ and $T(S) = V_i/V_o$. Hence $1/T(S)$ represents the desired voltage gain. Element 1 is made dependent on element 9 for this calculation through input H(T) = 1, which represents $1/T(S) = G(S)$ for this problem. All other entries in column H are coded zero.

The use of tagging parameters in the input serves to indicate quantities which require frequent and rapid identification. Column F lists the frequency dependence of the element E, tagging S^F where F = -1, 0, +1 for electrical networks. For example, a capacitor coded as an admittance has F = +1, since Y = SC. In this example all elements are frequency independent, hence F = 0 in all cases. The program accepts higher powers of S, hence is useful for electromechanical and mechanical systems.

Column K utilizes another tagging variable useful for sensitivity calculations. The entry K = 1 in the input for element 4 indicates to the program to calculate the sensitivity of the voltage gain to changes in the β of transistor 1, since element 4 is the controlled current generator of the transistor equivalent circuit.

Care must be exercised in the specification of numerical inputs to scale the value of an element according to its dichotomous description. For example, a resistor coded as an admittance must have as its numerical input a value of conductance rather than resistance. Dependent generators are coded with the value of the associated gain.

Construction of Flowgraph

Table 2 gives the output of the computer-constructed flowgraph in the form of three matrices V, W, and T, where N equals the number of elements; here N = 9. These outputs represent a judicious choice of the constraining equations for the closed system according to the tree chosen at the beginning of the problem formulation.

The matrices V and W define the Kirchhoff voltage and current restraints for the network, hence represent equations for network interconnections. The T matrix defines the set of differential equations for the network-intra-relationships for the elements.

For example from Table 2, reading down column 1 of the W matrix:

$$I(1) = -I(2) - I(3)$$

This is the Kirchhoff current constraint at node 1.

It is to be emphasized that these outputs are only an intermediate step in the problem solution and hence serve only a tutorial function here. They are not visually necessary for the flowgraph construction and evaluation, since this

is performed within core, and these intermediate results are stored and used within core.

Evaluation of Flowgraph

The basic concept in the evaluation of the flowgraph is the first order loop L (1,n). A first order loop is defined as a connected sequence of directed transmittances through the flowgraph which touches no node more than once. All n = 1, ---N first-order loops are consecutively evaluated by labyrinth routines and are tabulated with their symbolic and numerical values. The value of a loop is the product of its constituent transmittances.

A second order loop L(2,n) is defined as the product of two disjoint first order loops, i.e. those which have no common nodes. Similarly a loop of order m, L(m,n), consists of m disjoint first order loops. These loops are successively evaluated and presented in a similar manner to the first order loops.

For a given closed system a constraint exists among the loops of the system flowgraph. If there are N first order loops present, denote the sum of these loops by H(1) where

$$H(1) = \sum_{n=1}^{N} L(1,n) \qquad (2)$$

Let H(m) be the sum of all L(m,n). If the highest order loop present is of order M, then

$$H \triangleq \sum_{m=0}^{M} (-1)^m \; H(m) = 0 \qquad (3)$$

where H(0) = 1 and (3) is defined to be the topology equation constraint for closed systems which equals zero. H is thus one plus the signed sum of all loops present in the flowgraph.

The topology equation takes the form of a linear function of the fictitious closing parameter T,

$$H = H(\bar{T}) + T\,H(T') = 0 \qquad (4)$$

where H(T') is that part of the equation devoid of T

and H(T') is that part deprived of T (since T appears only linearly, it can easily be factored out by the computer using the tagging technique described)

Equation (4) can then be solved for the desired function G = 1/T yielding

$$G = 1/T = -H(T')/H(\bar{T}) \qquad (5)$$

which is the desired frequency-dependent transfer function.

Table 3 gives the computer flowgraph evaluation for example 1. The columns L(i) and H(i) give the number and order of the loops starting with H(0) numbered 0; E(i) defines which of the nine transmittances are contained in the loops. The columns T, K, and S indicate the tags for each loop which are formed as the algebraic sum of the tags of the constituent elements. Finally, the numerical value of each loop is listed, with $H(0) \triangleq 1$. The transfer function is then formed according to the procedure outlined above. The computer does this by forming separate signed sums according to (3) of those loops which are tagged with T = 0 or T = 1 since H(T) = H(T = 0) and H(T') = H(T = 1). The quotient is then formed according to (5) to yield Vg = 9.00. Table 4 gives the calculation of the sensitivity of the voltage gain with respect to β, which is discussed in more detail just below. For now, just note that this sensitivity = 1, showing that β has a direct effect on the voltage gain of the circuit.

DESIGN APPLICATIONS

Sensitivity Analysis

The flowgraph approach is particularly useful in the calculation of sensitivity functions. We define the sensitivity of a performance criterion P with respect to a parameter Q as

$$S_Q^P = \frac{d \log P}{d \log Q} = \frac{dP/P}{dQ/Q}$$

The program then calculates the symbolic and numerical sensitivity function defined above.

The details of the algorithm are not important here, but it is important to note that the technique eliminates the need for numerical schemes to take the required derivatives. These are accomplished instead by grouping algorithms for the symbolic topology equation. (13)

For the problem of example 1, the sensitivity of the voltage gain to variations in β is required. It can be shown that

$$S_{\beta_1}^{Vg} = \frac{\Sigma H(\bar{\beta}_1)}{\Sigma H(\bar{V}g)} = \frac{\Sigma H(K=0)}{\Sigma H(T=0)}$$

which can be obtained from Table 3.

$H(\bar{\beta}_1)$ and $H(\bar{V}g)$ are the parts of the

topology equation devoid of β_1 and Vg. A complete description of the sensitivity analysis portion of NASAP is available.

Transient and Frequency Response

A documented subroutine (14) is available to obtain the transient response of a network from the calculated transfer function. This subroutine produces the response of the network to impulse, step, ramp and sinusoidal inputs. An arbitrary input may be specified as an array of numbers. The outputs consist of a graphic display of the response as well as a table of numbers showing the response of the system at equally spaced intervals of time.

A plotting subroutine (Bode Plot) is also available to display the frequency dependent transfer function over a specified range of frequency giving both magnitude and phase.

Example 2: L-C Filter

To further illustrate the techniques used, and as an example of a circuit with frequency dependent elements, consider the L-C filter of Figure 4A. The dichotomized schematic is shown in Figure 4B while the flowgraph drawn from the computer output in Figure 4C is shown only as a visual aid. It is desired to find the input admittance of this circuit, and its sensitivity to variations in L2.

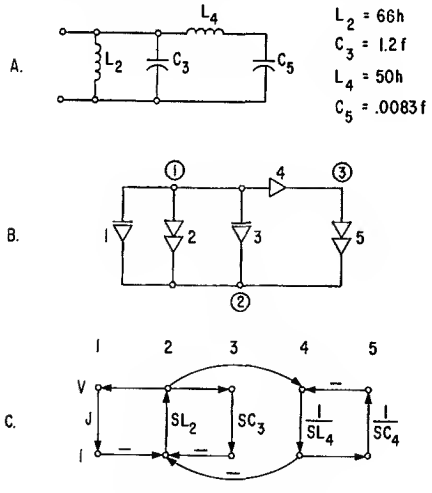


FIG. 4. Example 2: L-C Filter

The problem statement or input to the program is shown in Table 5, which has been formed according to the rules developed in the subsection on Coding of Equivalent Circuits for Computer Input. Note particularly row 1 of this array for which C = 0, D = 2, G = 1, and H = 1. This indicates that the unknown transmittance is one which relates the voltage generated by element 2 and the current generated by element 1: the input admittance. Column F of this matrix contains the appropriate entries for the frequency dependent elements. For example, element 3 is a capacitor coded as a voltage controlled current generator (C=0, G=1) or admittance, has frequency dependence of S^1 hence F (E=3) = +1. The K = 1 in row 2 indicates that the sensitivity analysis is to be performed with respect to L2.

Table 6 gives the flowgraph evaluation as before. The transfer function must now be formed according to the tags T=0 or 1, but also grouping those terms having the same S tag to form the appropriate coefficients of terms with like powers of S. An additional output here is the transfer function normalized as to the power of S and the coefficient of the highest power of S in both numerator and denominator polynomials. Table 7 gives the sensitivity analysis, also is a quotient of polynomials according to the previous subsection on Transient and Frequency Response.

It should be noted that these are not the only outputs available. Additional subroutines factor all polynomials of interest, form a transient response and a Bode plot. Planned sub-routines, as well as further work on the above routines are discussed in the concluding section.

CONCLUSIONS

The problem statement of NASAP seems at first appearance somewhat cumbersome to use, although with a little practice it becomes quite easy.

The choice of the tree at the very beginning of the coding seems to be quite critical to the running time of the problem. There is no theoretical information available on the best (least computing time) choice of a tree, but experience has shown that a closely connected tree (many voltage generators emanating from one node) has a shorter running time than one which is loosely connected. For most practical networks, the ground node naturally becomes this node.

The program at present handles a linear, time-invariant circuit with a maximum of 40 elements. Most practical networks can be adequately approximated by a linear equivalent circuit only in fairly narrow ranges of operation. For this reason it is planned to extend the program to piecewise linear and non-linear networks.

Flowgraph techniques may be used effectively to reduce the complexity of circuit models. An optimization technique is to be implemented as a subprogram which finds and eliminates all components which contribute to a desired result less than a pre-assigned value.

The NASAP program has received widespread attention because of its symbolically-oriented algorithms and its inherently logical output capabilities. In this respect it has fulfilled the initial reasons for its development as a research tool.

Grants and contracts are now in force to extend and develop NASAP into a useful design tool. The capacity is being extended to 120 elements. The input formats are being rewritten to provide automatic generation of the input matrix. But, since the construction of the matrix puts the designer in close touch with the circuit he is analyzing, it has been decided to leave this automatic generation as an option.

A complete plotting package is being written for display of the various outputs at the user's request. A tolerance analysis and a sensitivity matrix giving the sensitivity of transfer functions of interest to all components now available as subroutines, are being added modularly to the main program.

TABLE 1

Problem Statement for Two-Transistor Amplifier

A	B	C	D	E	F	G	H	K	Numerical
1	2	0	9	1	0	0	1	0	.100E+1
1	2	0	2	2	0	1	0	0	.100E-2
1	2	0	3	3	0	1	0	0	.100E-1
3	2	1	3	4	0	1	0	1	.100E+2
3	2	1	5	5	0	0	0	0	.100E+4
3	2	0	6	6	0	1	0	0	.100E-1
4	2	1	6	7	0	1	0	0	.100E+2
4	2	0	8	8	0	1	0	0	.100E-2
4	2	1	9	9	0	0	0	0	.100E+2

TABLE 2

Computer-Output-Flowgraph Construction

V	1	2	3	4	5	6	7	8	9
1		-	-						
2									
3									
4									
5				-		+			
6									
7									
8									
9							-	+	

W	1	2	3	4	5	6	7	8	9
1									
2	-								
3	-								
4					-				
5									
6					-				
7									-
8									-
9									

T	1	2	3	4	5	6	7	8	9
1									
2		Y							
3			Y	I					
4									
5					Z				
6						Y	I		
7									
8								Y	
9	E								Z

TABLE 3

Flowgraph Evaluation for Two-Stage Amplifier

L(i)	H(i)	E(i)	TKS ±	Numerical
0	0		000 +	.100E+1
1	1	1 34567 9	110 -	.100E+5
2	1	56	000 +	.100E+2
3	1	89	000 +	.100E-1
4	2	56 89	000 +	.100E+0

Transfer Function = .900E+1
Normalized Transfer Function = .900E+1

TABLE 4

Calculation of Sensitivity Function for Two-Stage Amplifier

	L(i)	Numerical	S
	0	.100E+1	0
$H(\bar{\beta}_1)$	2	.100E+2	0
	3	.100E-1	0
	4	.100E+0	0
		Sum = .111E+2	
	0	.100E+1	0
$H(\bar{V}g)$	2	.100E+2	0
	3	.100E-1	0
	4	.100E+0	0

Sum = .111E+2

Sensitivity = .100E+1

TABLE 5

Problem Statement For L-C Filter

A	B	C	D	E	F	G	H	K	Numerical
1	2	0	2	1	0	1	1	0	.100E+1
1	2	1	2	2	1	0	0	1	.666E+1
1	2	0	3	3	1	1	0	0	.120E+1
1	3	0	4	4	-1	1	0	0	.200E-1
3	2	1	5	5	-1	0	0	0	.120E+3

TABLE 6

Flowgraph Evaluation For
L-C Filter

L(i)	H(i)	1 2 3 4 5	T K S +	Numerical
0	0		0 0 0 +	.100E+1
1	1	1 2	1 1 1 +	.666E+0
2	1	2 3	0 1 2 +	.880E+0
3	1	2 4	0 1 0 +	.133E-1
4	1	4 5	0 0-2 +	.240E+1
5	2	1 2 4 5	1 1-1 +	.160E+1
6	2	2 3 4 5	0 1 0 +	.192E+1

$$\text{Transfer function} = \frac{.66S + 1.6S^{-1}}{.880S^2 + 2.9 + 2.4S^{-2}}$$

$$\text{Normalized Transfer function} = .75\,\frac{S^3 + 2.42S}{S^4 + 3.31S^2 + 2.72}$$

TABLE 7

Sensitivity of Voltage Gain of L-C Filter
to Variations in L_2

	L(i)	Numerical	S
$H(\bar{L}_2)$	0	.100E+1	0
	4	.240E+1	-2
Sum = .100E+1 + (.240E+1)S^{-2}			
$H(\bar{Z}in)$	0	.100E+1	0
	2	.880E+0	2
	3	.133E-1	0
	4	.240E+1	-2
	6	.192E+1	0

Sum = (.880E+0)S^2 + 2.9 + (.240E+1)S^{-2}

$$\text{Sensitivity} = 1.14\,\frac{S^2 + 2.40}{S^4 + 3.31S^2 + 2.72}$$

REFERENCES

1. R. M. Carpenter and W. W. Happ, Algorithms for the Dichotomous Representation of Macrocircuits, Proceedings, Fourth Annual Allerton Conference on Circuit and System Theory, October, 1966. Accepted for publication by IEEE Transactions on Systems, Science, and Cybernetics, for Aug., 1967.
2. R. M. Carpenter, Topological Analysis of Active Networks, Proceedings, Institute on Modern Solid State Circuit Design, University of Santa Clara, Santa Clara, California, Sept., 1966.
3. R. M. Carpenter, Algorithms for the Construction and Display of Flowgraphs, Proceedings, Second NASA Microelectronics Conference, Washington, D. C., Sept., 1966.
4. R. M. Carpenter and W. W. Happ, Symbolic Approach to Computer-Aided Circuit Design, Electronics, Vol. 39, No. 25, pp. 92-98, December, 1966.
5. W. W. Happ, Flowgraph Techniques for Closed Systems, IEEE Trans. AES-2, No. 3, pp. 252-264, May, 1966.
6. W. W. Happ, Flowgraphs as a Teaching Aid, IEEE Trans. E-9, No. 2, pp. 69-79, June, 1966.
7. W. W. Happ and J. Staudhammer, Topological Techniques for the Derivation of Models with Pre-Assigned Accuracy, Archiv der Elektrischen Ubertragung, Heft 6, 329-335, A.E.U. 20, June, 1966.
8. R. M. Carpenter and W. F. Hartman, Desensitizing of Microcircuits to Variations in Temperature and Production Spread; Sensitivity, Proceedings, Second NASA Microelectronics Conference., September, 1966.
9. L. Hasdorff and E. V. Harrington, Jr., A Computer-Oriented Approach to Coding, Evaluating and Optimizing Acoustic Systems, NASA Report, ERC/CQ, October, 1966, to be submitted to the Journal of the Acoustical Society of America.
10. L. Hasdorff and E. V. Harrington, Computer-Oriented Approach for Coding, Evaluation and Optimizing Electrical-Mechanical, Hydraulic Systems, NASA Report ERC/CQ 66-657, June, 1966, to be submitted to IEEE Trans. Ed.
11. W. F. Rombalski and R. M. Carpenter, Desensitizing of Microcircuits to Variations in Temperature and Production Spread, Phase IV, Determination of the Effect of Production Spread of Parameters, Proceedings, Second NASA Microelectronic Conference, September, 1966.

12. W. W. Happ, E. R. Robbins, and D. E. Moody, Modeling Piecewise-Linear Systems by Topological Techniques, IEEE Trans. AES., Vol. AES-2, No. 4, pp. 370-375, July, 1966.

13. R. M. Carpenter, Computer-Aided Sensitivity and Tolerance Techniques, Proceedings, IEEE/ESD Symposium, Boston, April, 1967. Also to be submitted to IEEE - Trans. Ct., for July, 1967.

14. L. Hasdorff and W. I. Grosky, Numerical Computation of the Time Responses of a System from its Transfer Function, NASA Report ERC/CQ 66-659, August, 1966, submitted to IEEE Trans. AES.

A SURVEY OF TRANSIENT CIRCUIT ANALYSIS COMPUTER PROGRAMS

By

CAPT. GARY K. PRITCHARD

Capt. Pritchard is with the United States Air Force, Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico. He received a B. S. E. E. degree from the University of Florida in 1962, and a M. S. degree in Electrical Engineering from the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, in 1964.

Since 1964 he has been assigned to the Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico, where he has been working in the area of transient radiation effects on electronics. This work involved the development of non-linear transient circuit analysis computer programs and their application to the prediction of transient radiation effects on electronic circuits. He is the project officer for the PREDICT and SCEPTRE transient analysis programs.

Capt. Pritchard is a member of Phi Kappa Phi, Tau Beta Pi, Eta Kappa Nu, and the Institute of Electrical and Electronic Engineers.

A SURVEY OF TRANSIENT CIRCUIT ANALYSIS PROGRAMS

By Gary K. Pritchard
Captain, USAF
Project Officer
Air Force Weapons Laboratory
Kirtland AFB, New Mexico

SUMMARY

The basic features of ten automated nonlinear transient circuit analysis programs are listed and the values of each of the features discussed from a potential user's viewpoint. The features compared for each program include the types of analysis performed, the types of elements handled, built-in models, the computers on which the programs are operational, documentation, program availability, user convenience, special features, and mathematical solution formulation.

INTRODUCTION

This paper compares the major features of several digital computer programs for nonlinear transient circuit analysis. The features of the programs are presented from the user's viewpoint and include discussions of the capabilities, limitations, and availability of each program.

There are in existence a number of transient circuit analysis computer programs. Most of these programs can be applied to Transient Radiation Effects on Electronics (TREE) problems. Since there is no current, comparative documentation on the capability, limitations, and availability of these programs, the Defense Atomic Support Agency (DASA) requested the Air Force Weapons Laboratory (AFWL) review the programs in existence.

APPROACH

This survey complements earlier surveys by Wirth[1] in 1964 which briefly compared PREDICT, NET-1, CIRCUS, MISSAP, and ECAP; by Dickhaut[2] in 1965 which compared PREDICT, NET-1, and CIRCUS; and by Pritchard[3] in 1965 which compared TRAC with PREDICT, NET-1, and CIRCUS. Programs reviewed previously have been modified, new programs have been developed, and several other industry- and university-developed programs have become of general interest through recent listings in technical journals.[4,5]

Information for this survey was primarily gathered from a questionnaire which was sent to sixteen different government laboratories, corporations, and universities which were known to have developed sophisticated circuit analysis programs.

The questionnaire was formulated to gather information in a concise form about all aspects of the program of interest to a potential user of the program. The questionnaire consisted of seven sections:

I. Capability of the Program: This section was concerned with the types of analysis performed (ac, dc, transient), kinds and number of elements handled, and built-in model capability.

II. Computer Compatibility: This section asked on which computers the program was operational, in what language the program was written, the size memory required, and questions concerning the status and documentation of the program.

III. Radiation Effects: This section was concerned with special features of a program for handling radiation effects problems.

IV. Use, Convenience, and Flexibility Factors: This section was concerned with input and output formats, restrictions on circuit topology specification, and general features such as rerun options.

V. Mathematical Details: This section was concerned with the types of solution formulation and integration routines used in the program.

VI. Availability: This section was concerned with the availability of the program and the procedure for obtaining the program.

VII. Additional Information: This section asked for a user's manual and a sample problem run for each program.

RESULTS

From the information obtained from the questionnaires,* ten programs are categorized as automated nonlinear transient circuit analysis programs. These programs are automated in that they formulate the circuit equations from topological and component data and perform the solution calculations to provide a transient or time history analysis of the circuit. The programs are also powerful enough to include nonlinear elements (capacitors, resistors, inductors, or voltage or current sources) used to model active electronic devices.

The names of the ten programs included in this survey are given in Table I along with the names of the originating organization and

*See Acknowledgments section of this paper and references 6 through 13.

government sponsor. A comparison of the major features and capabilities of the programs is shown in Tables II through IV.

DISCUSSION

The program features are compared in Tables II through IV. The following is a user-oriented discussion of these features. As seen in Table II all programs except PREDICT provide a steady-state dc solution. Each program uses an iterative technique, usually a modified Newton-Raphson to obtain the dc solution. This provides an efficient means of obtaining the initial conditions required for a subsequent transient analysis. Two of the programs, the General Network Analysis program and the MISSAP I also provide nominal ac solutions and frequency or Fourier analysis.

One of the first concerns of a transient analysis is the initial conditions of the circuit. For all programs, except PREDICT, the initial conditions for the transient analysis can be automatically calculated by the dc solution portion of the program. For PREDICT the dc solution can be obtained by a separate "power supply turn-on" transient run. All of the programs except CIRCUS and the General Network Analysis program provide for entering user-supplied initial conditions. This feature is particularly useful for circuits which the dc iterative solution fails to converge and the circuit initial condition must be approximated from measurements or separate transient runs.

An important feature of each program is the type of elements or components which may be entered. All of the programs readily accept constant valued resistors, capacitors, inductors, and constant and time-varying voltage sources. All except NET-1 also include constant and time-varying current sources. All except TAG and TRAC provide for entering inductive coupling through mutual inductance. These elements along with built-in models for active devices provide an adequate tool for describing most standard discret component transistorized circuits. For applications such as modeling new semiconductor devices, more flexibility is required. For these applications it is important to be able to define functionally-variable elements and enter these into the program. Several of the programs provide for this by allowing variable elements defined by a table (F_2), equation (F_3), or subroutine (F_4), as shown in Table II.

An indication of the maximum size of the circuit which may be entered is shown in the next two columns of Table II. This varies from 30 nodes for the General Network Analysis program and 60 elements for the Oklahoma State Analysis program, to 300 nodes for the SCEPTRE program and 600 elements for the NET-1R program. For some of the programs, additional limits are placed on the number of each type of element, while others limit only the total number of elements or nodes.

The CIRCUS, General Network Analysis, NET-1R, and TRAC programs include fixed built-in models for active devices. These models are considered fixed models in that they cannot readily be changed by a user. The built-in transistor and diode models of these programs are nonlinesr transient Ebers-Moll models and are essentially the same for each of these programs.

MISSAP III built-in models are defined in subroutines which can be changed by a user. Other programs, PREDICT, SCEPTRE, STRAP, and TAG do not include built-in models. For these programs active devices are entered by user-defined equivalent circuit models either as part of an overall circuit, or in the case of SCEPTRE, from a user-defined model library tape.

The last column on Table II indicates whether models or model parameter data may be stored and called from e model library tape. The SCEPTRE-stored model feature is unique in that the stored model is user defined and may have up to 25 external terminals.

Table III indicates the computers on which each of the programs are operational, the language it is written in, the extent of the documentation, and the availability of each program.

If a program is capable of solving a particular problem and is operational on an available computer, whether or not the program is used may be determined by how difficult the program is to use. All of the programs have been designed to be used by engineers without requiring the services of a programmer. All of the programs use an engineer-oriented input language. In general, the less restrictive, easier to use programs provide for a free-field input as indicated in Table IV.

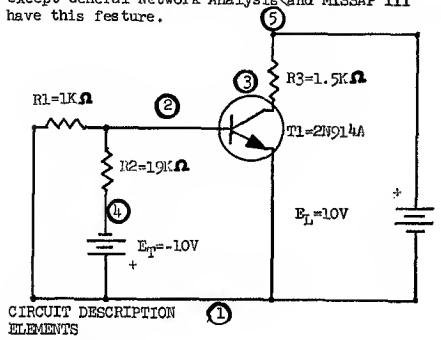
An example free-field SCEPTRE input is shown in Figure 1. The input for this circuit would look similar for CIRCUS, with commas replacing the dash and equal signs, and for NET-1 with blanks replacing the commas, dashes, and equal signs. The input for PREDICT would be similar except the transistor would have to be specified as equivalent circuit elements and defining equations. STRAP would also appear similar except variable element values would be entered in a subroutine.

An example of the free-field subroutine format of MISSAP III is shown in Figure 2.

TAG also uses a free-field subroutine format which is flexible, but less convenient to use. TRAC uses a fixed field format for element specification plus the use of auxiliary FORTRAN equations for defining nonstandard elements or outputs.

An economically important feature is a save and continue feature that permits the user to save the results of a transient run in a form that permits the run to be continued from that point. This is particularly valuable for analyzing large circuits which may require several minutes of computer time for a few microseconds of problem time and where the circuit recovery

time cannot be estimated accurately. All programs except General Network Analysis and MISSAP III have this feature.



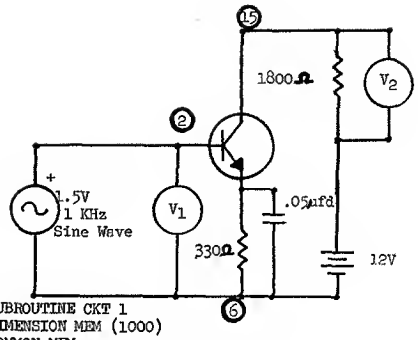

```
CIRCUIT DESCRIPTION
ELEMENTS
R1, 1 - 2 = 1
R2, 2 - 4 = 19
R3, 5 - 3 = 1.5
E1, 1 - 5 = 10
ET, 4 - 1 = 10
T1, 2 - 1 - 3 = MODEL 2N914A (PERM)
OUTPUTS
VR3, VR1, VCXT1, PLOT
RUN CONTROLS
STOPTIME = 500
RUN INITIAL CONDITIONS
END
```

Figure 1, Example of SCEPTRE Input Format

Another convenient feature of most of the programs is one that permits the circuit analysis to be rerun automatically, with circuit parameter changes (element value, transient forcing function, etc.) by requiring only specification of the changes.

For all the programs, transient output is a time history of the circuit response. Some of the programs such as CIRCUS, General Network Analysis, and NET-1R essentially provide only node voltages and semiconductor junction currents and voltages as output. PREDICT provides only element currents and voltages as output. Other programs such as SCEPTRE, TAG, and TRAC also provide for additional user-defined outputs which are combinations of circuit variables. Using this feature, quantities such as element power dissipation and voltage between arbitrary points in a circuit are readily available for output. MISSAP III provides a unique method for specifying output by inserting voltmeters and ammeters in the circuit and then printing these "meter readings" as output.

Most of the programs provide for plotted results as well as printed listings as shown in the "plots" column of Table IV. Some of the programs such as CIRCUS and SCEPTRE provide plot information which can be processed by user installation supplied plot routines to obtain additional plotted results.




```
SUBROUTINE CKT 1
DIMENSION MEM (1000)
COMMON MEM
CALL MISSAP (61, 40, 1000)
CALL E (2, 6, 3, 1.5, 0, 1.0E-3, 1000., 0)
CALL V (2, 6, 1, 1, 1.0E-3)
CALL T (2, 4, 2, 15, 1.0E-6, 0.98, 1.0E-7, 39.5)
CALL R (4, 6, 330.0)
CALL C (4, 6, 0.05E-6, 0.7, 0)
CALL R (15, 8, 1800.0)
CALL DC (8, 6, 12.0)
CALL V (15, 8, 2, 1, 1.0E-3)
CALL QRUN (0, 5.0E-3, 5.0E-4)
RETURN
END
```

Figure 2, Example of MISSAP III Input Format

The next four columns of Table IV indicate the solution termination options available for each program. Most of the programs provide for solution termination when the problem response time limit specification is reached, when a specified computer machine time limit is reached, and when the solution time step increments decrease below a minimum limit. MISSAP III, SCEPTRE, and TAG also provide for termination when circuit variables exceed user-specified limits.

One of the major uses of these programs is providing circuit nuclear radiation response calculations. All but three of the programs, MISSAP III, Oklahoma State Systems Analysis, and TAG were either designed or have been adapted specifically for radiation effects applications.

Some of the programs provide directly for radiation effects analysis as shown in Table IV by including photocurrent generators in the built-in semiconductor device models. These programs also usually provide a simple format for entering radiation effects data. These programs also provide for entering or calculating radiation induced device model parameter changes.

The remaining columns of Table IV indicate the basic approach used for the equation formulation and solution by each of the programs. For the transient solution either a state variable (capacitor voltage and inductor current) or nodal (admittance matrix) formulation is used. Generally, an explicit numerical integration routine

is used with the state variable formulation. Generally, for the nodal formulation, implicit integration by including recursive difference approximations in the admittance matrix is used for solution with an accompanying iteration solution technique for nonlinearities and dependent sources at each solution time increment.

There are other important aspects of these programs that have not been included in this paper. One is the relative solution efficiency of the various programs. Previous comparisons have shown differences of up to a factor of ten in the computer time required for different programs to calculate the same circuit response on a similar computer. Although efficiency is important, its accurate evaluation requires that all programs be run on the same computer for a family of circuit problems.

CONCLUSIONS

The programs reviewed were quite similar in capability of analyzing most transistorized circuits. The entering of most circuit data is convenient and quite similar for CIRCUS, NET-1R, and SCEPTRE, with TAG and TRAC probably the least convenient.

Considerable differences in the flexibility of the programs were noted. CIRCUS, General Network Analysis, and NET-1R programs are fairly inflexible in that only fixed value elements are permitted outside of the built-in models. The other programs are capable of processing functionally variable elements and so are useful for performing modeling work. However, the ease of entering variable elements varies considerably between programs. Only PREDICT, SCEPTRE, and STRAP make provisions for describing variable elements in simple engineer-oriented language.

ACKNOWLEDGMENT

The author wishes to thank John Hubbard and John Anderson for their assistance in preparing the survey questionnaire and the following for their efforts in completing the questionnaire and providing information about their respective programs: R. H. Dickhaut (CIRCUS), J. J. Menard (General Network Analysis program), R. J. Reid (MISSAP III), H. Puttcamp (NET-1R), J. Waldon (Oklahoma State Systems Analysis program), S. R. Sedore (SCEPTRE), K. A. Wyse (STRAP), W. K. Shubert (TAG), E. D. Johnson and C. Kleiner (TRAC).

REFERENCES

1. Wirth, J. L., "The Design and Analysis of Electronic Circuits by Digital Computers," SCR-64-1355, Sandia Corporation, October 1964.

2. Analytical Methods and Fundamental Parameters for Predicting Responses of Electronic Circuits to Transient Nuclear Radiation, with Application to Hardened Circuit Design, AFWL TR 65-105, Air Force Weapons Laboratory, July 1965.

3. Memorandum, Pritchard, G. K., Air Force Weapons Laboratory, to Ballistic Systems Division, Evaluation of TRAC Computer Code (unpublished).

4. Falk, Howard (Editor), "Computer Programs for Circuit Design," Electro-Technology, Vol. 77 No. 6, Jun 1966, pp. 54-57.

5. Dunanian, John, "Check Design Program Availability," Electronic Design, Vol. 14, No. 23, 11 October 1966, pp. 76-80.

6. Milliman, L. D., Massena, W. A., Dickhaut, R. H., User's Guide to CIRCUS, to be published, 1967.

7. Perlich, D. R., and Kimball, E. M., Utilization Manual Computer Program 2M23, General Network Analysis Nonlinear Steady-State and Nonlinear Transient Programs, Lockheed Misaile and Space Company.

8. The Michigan State System Analysis Program, MISSAP I, Michigan State University, East Lansing, Michigan, 15 May 1966.

9. MISSAP III, General Information Manual, Electronic Circuit Analyses, Michigan State University, East Lansing, Michigan, 25 July 1966.

10. Malmberg, A. F., Cornwell, F. N., NET-1 Network Analysis Program, 7090/7094 Version, LA-3119, Los Alamos Scientific Laboratory, Sept 1964.

11. Automated Digital Computer Program for Determining Responses of Electronic Systems to Transient Nuclear Radiation, PREDICT Circuit Analysis Program, WL TDR 64-62, Vol II, Air Force Weapons Laboratory, August 1964.

12. Automated Digital Computer Program for Determining Responses of Electronic Circuits to Transient Nuclear Radiation (SCEPTRE), Vol. I, SCEPTRE User's Manual, Vol. II, SCEPTRE Formulation, PREDICT Support, AFWL TR 66-126, Air Force Weapons Laboratory, February 1967.

13. King, K. O., Gillet, K., Libaw, W., TAG User's Manual, R-847, Planning Research Corporation, June 1966.

TABLE I--AUTOMATED NONLINEAR TRANSIENT CIRCUIT ANALYSIS PROGRAMS, ORIGINATORS, AND SPONSORS

Program	Originator	Sponsor
CIRCUS (Circuit Simulator)	Boeing Company	
General Network Analysis Program	Lockheed Sunnyvale, Calif	US Navy
MISSAP III (Michigan State Systems Analysis Program)	Michigan State Univeraity East Lansing, Michigan	
NET-1R (Network Analysis Program, Radiation Version)	Los Alamos Scientific Laboratory and Braddock, Dunn, & McDonald, Inc	LASL (AEC) HDL, US Army
Oklahoma State Systems Analysis Program	Oklahoma State University Stillwater, Oklahoma	
PREDICT (Prediction of Radiation Effects by Digital Computer	IBM Corporation Owego, New York	AFWL, USAF
SCEPTRE (System for Circuit Evaluation and Prediction of Transient Radiation Effects)	IBM Corporation Owego, New York	AFWL, USAF
STRAP (Simplified Transient Radiation Analysis Program)	Douglas Aircraft Co. Santa Monica, Calif	
TAG (Tranaient Analyais Generator)	Jet Propulsion Laboratory Pasadena, Calif	NASA
TRAC (Tranaient Rediation Analyais by Computer)	Autonetics Anaheim, Calif	

TABLE II: MAJOR FEATURES AND CAPABILITIES

Program	D.C. Analysis					Transient Initial Conditions		Types of Elements								Built-in Models								
	Initial Conditions	Sensitivc Coefficient	Component Variation	A.C. Analysis	Transient Analysis	Automatically Calculated	Entered by User	Resistors	Capacitors	Inductors	Mutual Inductance or Coefficient of Coupling	Voltage Source	Current Source	Maximum No. Elements	Maximum No. Nodes	Transistor	Diode	Tunnel Diode	Zener	SCR	FET	Magnetic Core	Fixed Model	Stored Model
CIRCUS	X		X		X	X		C	C	C	C	C,F_1	C,F_1	400	200	X	X	X		X			X	X
GEN NET-WORK ANAL PROGRAM	X	X	X	X	X	X		C	C	C	C	C,F_1	C,F_1		30	X	X	X	X		X		X	X
MISSAP III	X		X	X*	X	X	X	C,F_4	C,F_4	C,F_4	C	C,F_{14}	C,F_{14}	** 100		X	X							X
NET-1R	X				X	X	X	C	C	C	C	C,F_1		600	100	X	X						X	X
OKLAHOMA STATE ANAL PRO	X				X	X	X	C,F_3	C,F_3	C,F_3	C	C,F_{13}	C,F_{13}	60		X	X							
PREDICT					X		X	C,F_{123}	C,F_{123}	C,F_{123}	C	C,F_{123}	C,F_{123}	300	100									
SCEPTRE	X		X		X	X	X	C,F_{1234}	C,F_{1234}	C,F_{1234}	C,F_{134}	C,F_{1234}	C,F_{1234}	300	300		X							X
STRAP	X				X	X	X	C,F_{123}	C,F_{123}	C,F_{123}	C,F_{123}	C,F_{123}	C,F_{123}	300	100		X							
TAG	X		X		X	X	X	C,F_{1234}	C,F_{1234}	C,F_{1234}	***	C,F_{1234}	C,F_{1234}		100									
TRAC	X		X		X	X	X	C,F_{134}	C,F_{134}	C,F_{134}		C,F_{134}	C,F_{134}	100	60	X	X					X	X	

* MISSAP I
** In state vector
*** Ideal transformer

Element Values
C = Constant value.
F = User-defined variable value
F_1 = Function of time
F_2 = Function of circuit variables defined by a table
F_3 = Function of circuit variables defined by an equation
F_4 = Function of circuit variables defined by a subroutine

TABLE III--COMPUTER COMPATIBILITY, DOCUMENTATION, AND AVAILABILITY

Program	Operational on Following Computers	Language and Size of Source Deck	Minimum Memory Words (Decimal)	Documentation			Program Availability		
				User's Manual	Math Description	Program Org	Readily Available (no cost)	Special Restrictions	Request Through
CIRCUS	IBM 7094, SRU 1107/1108, GE 625/635, CDC 6600	98% FORTRAN IV 5500 cards	32K	X	X	X	X		R. Puttcamp, Harry Diamond Labs, Wash, DC
GEN NET-WORK ANAL PRO	SRU 1107/1108	100% FORTRAN IV 3000 cards	65K	X				Available to Government	Director, Specisl Projecta, Code SP 2702, Wash, DC
MISSAP III	CDC 3600	99.5% FORTRAN IV, 2000 cards	32K	X				Negotiated Contract	R. J. Reid, Michigan State Univ, East Lansing, Mich
NET-1R	IBM 7094, IBM 7040/7044, MANLAC II	100% FAP 30,000 cards	32K	X*			X		R. Puttcamp, Harry Diamond Labs, Wash, DC
OKLA ST. ANAL PRO	IBM 1410 (IBM 7040 version planned)	100% FORTRAN IV, 2000 cards	40K		X			Small cost IBM 7040 version to be available	J. Walden, Okla State University, Stillwater, Okla
PREDICT	IBM 7090/7094	50% FORTRAN II 50% FAP, 10,000 cards	32K	X	X	X	X		G. Pritchard, Air Force Weapons Lab, Kirtland AFB, New Mexico
SCEPTRE	IBM 7090/7094(CDC 6600 & IBM 360 versions planned)	100% FORTRAN IV, 15,000 cards	32K	X	X	X	X		G. Pritchard, Air Force Weapons Leb, Kirtland AFB, New Mexico
STRAP	IBM 7090/7094	100% FORTRAN IV	32K	X **	X	X		Negotiated Contract (Undefined)	G.C. Litman, Douglas Acft, Santa Monica, Calif
TAG	IBM 7090/7094	98% FORTRAN II 12,000 cards	32K	X	X		X		W.J. Thomas, Jet Propulsion Lab, Pasadena, Calif
TRAC	IBM 7090/7094	95% FORTRAN II 1500 cards	32K	X	X	X		Availability Undefined	C. Kleiner, Autonetics, Anaheim, Calif

* NET-1 only
** Being published

TABLE IV: CONVENIENCE, SPECIAL FEATURES, AND MATHEMATICAL DETAILS

Program				Outputs				Termination				Radiation		D.C. Solution		Solution Formulation					
	Free Field Input	Save and Continue	Rerun with Changes	Node Voltage	Element Current Voltage	Junction Current Voltage	Plots	Problem Time	Computer Time	Minimum Step	Circuit Response	Photo Current	Parameter Permanent Changes	Iterative (Newton-Raphson)	Integration	Explicit Integration	Implicit Integration	Iterative	State Variable	Modal	Automatically Variable Time Step
CIRCUS	X	X	X	X		X		X	X	X		X	X	X		X			X		X
GEN NETWORK ANAL PRO				X		X	*	X	X			X	X	X	X		X	X		X	
MISSAP III	X		X	X	X	X	***	X	X	X	X			X	X		X	X	X		X
NET-1R	X	X		X		X		X				X	X	X		X			X		X
OKLA STATE ANAL PRO	X+	X+	X		X	X		X	X					X		X		X	X		X+
PREDICT	X	X			X		**	X	X	X						X			X		X
SCEPTRE	X	X	X	#	X	X	***	X	X	X	X			X		X			X		X
STRAP	X	X	X	X	X	X	***	X				X	X	X			X			X	X
TAG	X	X	X	#	#	#	*	X	X	X	X			X		X		X	X		X
TRAC		X		X	X#	X	* ***	X	X	X		X	X	X			X	X		X	X

* SC 4020
** Cal Comp
*** Printer Plots

\+ IBM 7040 Version
\# User Defined Output Available

A USER'S VIEW OF ECAP

By

LEONARD DANKER

Bendix Radio Corp., Baltimore, Md.

Mr. Danker is a Scientific Engineering Programmer for the Bendix Radio Division of the Bendix Corporation. His responsibilities are the development of computer aided circuit design and analysis programs. Formerly a test equipment engineer, he has had experience with the IBM System 360 - Model 50, IBM 1401 and CDC G-20 computers.

Mr. Danker was awarded a B. S. degree in Electrical Engineering in 1965 from the City College of New York.

1620 CARDS - BCD CODE

The circuit cards that have a BCD Code can still be used with System 360 if a card with the following characters is inserted in front of the circuit deck.

* B C D Columns 1-4

All the following circuit decks will be assumed in the BCD mode unless a new mode card of the following form is encountered.

* E B C D I C Columns 1-7

This card returns the system to the EBCDIC 360 mode.

DC - AC TRANSIENT OPTIONS

The following calculation can be made in all three types of analysis:

		CODE
1.	NODE VOLTAGES	NV or Voltages
2.	ELEMENT CURRENTS	CA or Currents
3.	ELEMENT VOLTAGES	CV
4.	BRANCH CURRENTS	BA
5.	BRANCH VOLTAGES	BV
6.	ELEMENT POWER LOSS	BP

1 ERROR

The control card 1 ERROR = P is available in all three analyses.

A summation is made of the currents at all nodes and if the SUM of the unbalances is greater than the 1 ERROR value the unbalances and the branch currents are printed.

DC ANALYSIS OPTIONS

I. Partial Derivatives and Sensitivity Coefficients

The partial derivatives refer to the rate of change of voltage at a particular node with respect to a circuit parameter in a particular branch. The four partial derivatives available in DC analysis are the following:

i - node j - branch

1. $\frac{\partial E_i}{\partial R_j}$ R_j is a Resistor in branch j

2. $\frac{\partial E_i}{\partial E_j}$ E_j is a voltage source in branch j

3. $\frac{\partial E_i}{\partial I_j}$ I_j is a current source in branch j

4. $\frac{\partial E_i}{\partial GM_j}$ GM_j is transconductance in branch j

The sensitivity coefficients are defined as the change in node voltage for a one-percent change in the branch parameter. The four sensitivity coefficients available in DC analysis are the following:

i - node j - branch

1. $\frac{\partial E_i}{\partial R_j} \times \frac{R_j}{100}$

2. $\frac{\partial E_i}{\partial E_j} \times \frac{|E_j|}{100}$

3. $\frac{\partial E_i}{\partial I_j} \times \frac{|I_j|}{100}$

4. $\frac{\partial E_i}{\partial GM_j} \times \frac{GM_j}{100}$

II. Worst Case Solution

Tolerance data must be supplied in the following manner for a worst case solution:

R = 1000. (900., 1100.) or R = 1000.(.1)

E = 20. (19., 21.) or E = 20.(.05)

The worst case maximum solution is defined as the sum at the k^{th} node as

$E_K + \sum_i \frac{\partial E_K}{\partial P_i} \Delta P_i$ E_K - nominal value.

where all the terms $\frac{\partial E_K}{\partial P_i} \Delta P_i$ are positive.

P_i - Nominal value of parameter

ΔP_i - Tolerance of the parameter

If a worst case minimum solution is desired, all the terms are negative.

After the nominal solution has been calculated, parameter values are chosen for the worst case minimum solution. For the minimum solution, the minimum value of a

parameter ie chosen if ite partial derivative is positive. If the partial derivative is negative, the maximum value ie used. A new solution for node voltage ie made using the new parameter valuee. The partial derivative is calculated again, and if a sign change from the nominal case is encountered, a warning message is printed out. This warning message indicates that a true worst case minimum has not been found.

Next, the worst case maximum solution is obtained. The procedure is the same ae that of the minimum solution except that if the partial derivative is positive, the maximum parameter value is substituted and if the partial derivative is negative, the minimum parameter value is substituted. Again, if a sign change in the partial derivative is encountered, a warning message is printed out and the true worst case maximum has not been found.

III. Standard Deviation

The calculation of etandard deviations requires that the user supply the ± 3 standard deviation values of the circuit parametere. These values are entered the same way that tolerance data ie entered. An assumption is made that the circuit output variables are linearly related to the parameters, so that the partial derivatives are nearly constant over the range between maximum and minimum values of the parameters.

DC-AC OPTIONS

Parametric Variation

Parameters can be varied only in DC and AC analysis. For example, to change a branch that contained a resistor

```
B1  N(0,1), R = 10
```

another two cards of the following form would be placed after the last branch card.

```
B 13 N(5,0), R = 500  LAST BRANCH CARD
     MODIFY
B 1  R = 5(3) 20
     PRINT, VOLTAGES
     END
```

The resistor in branch 1 would be incremented from 5 to 20 ohms in 3 steps (5Ω per step). The node voltages would be printed out for each increment.

A single value could have been substituted in the following manner

```
     MODIFY
B1   R = 20
```

Miscellaneous

The MISCELLANEOUS output block indicator can be used in DC and AC analysis only.

When MISCELLANEOUS is encountered in a DC analysis print command, the nodal admittance matrix, equivalent current vector, and nodal impedance matrix will be printed.

In an AC analysis, only the nodal admittance matrix and the equivalent current vector will be printed if MISCELLANEOUS ie encountered in a print command.

AC OPTIONS

Frequency Variation

Frequency variation can be either linear or geometric. For linear frequency variation, the following two cards are entered

```
MODIFY
FREQ = P1 (+P2)P3
```

P1 - starting frequency

P3 - end frequency

P2 - number of eteps

For geometric frequency variation, the following two cards are entered

```
MODIFY
FREQ = P1 (P2)P3
```

P1 and P3 are the same as above

P2 - multiplier

TRANSIENT ANALYSIS OPTIONS

Time Dependent Voltage and Current Sources

The card containing the source information goes behind the branch that the source is located in. The card format is the following

Xnn (K), P0, P1, P2, ... Pn.

X can be E or I nn = branch number

K = TIME STEPS the source value is specified.

P0, P1, P2, ...Pn = source values at selected time intervals.

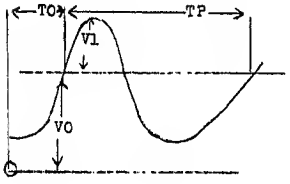
Periodic Source

Xnn P(K), P0, P1, P2, ... Pn

↖ indicates a periodic source

Sinusordal Source

Xnn SIN (TP), V1, V0, T0



Equilibrium

If an EQUILIBRIUM control card is placed after the print statement, only the steady state solution will be printed out.

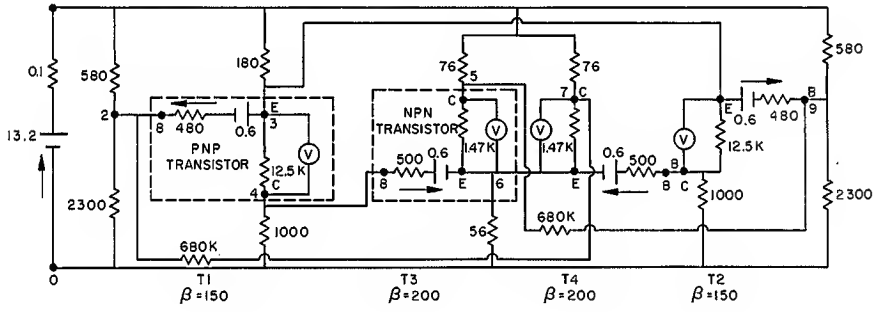
2 ERROR

The 2 ERROR = P card determines the time within which switch actuation takes place. The actuation will take place in P x TIME STEP of the TIME STEP in which actuation occurs.

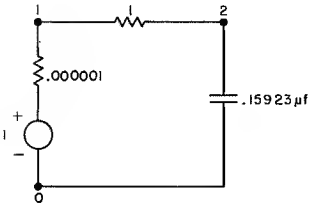
3 ERROR

The card 3 ERROR = P can reduce the time step after an initial condition solution. If P > 1, no reduction of the time step takes place. If P < 1, then the first time step immediately after the initial condition solution, is sub-divided into four parts.

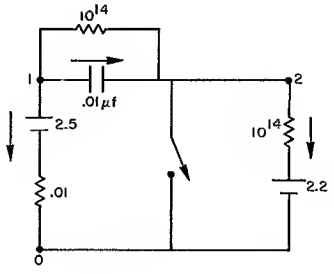
CIRCUIT 1



CIRCUIT 2
LOW PASS FILTER



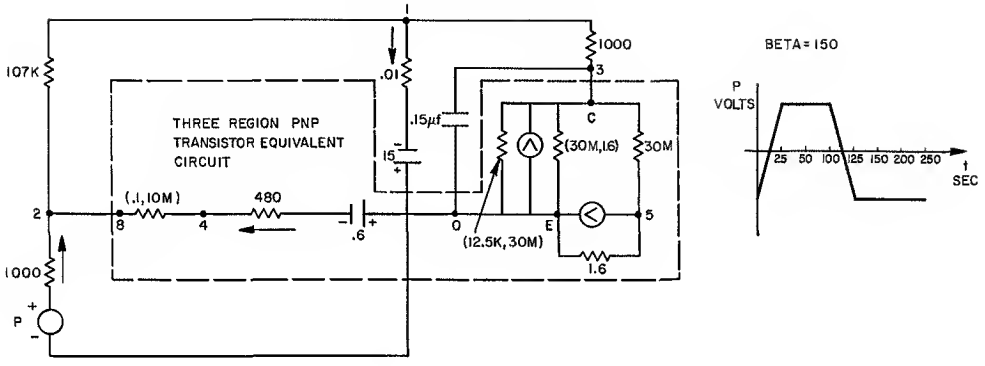
CIRCUIT 3



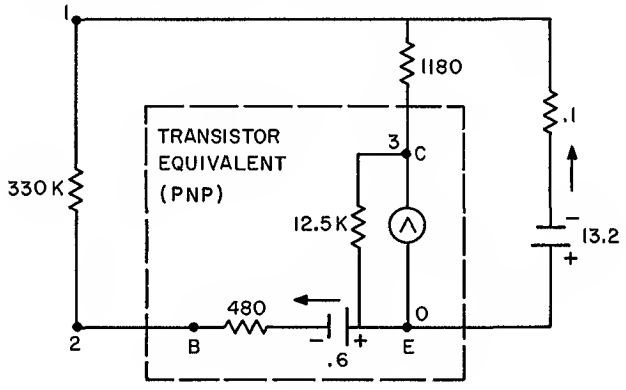
SWITCH IS OPENED AT T = 0

$V_{12}(0-) = 2.5$

CIRCUIT 4
TRANSISTOR SWEEP CIRCUIT



CIRCUIT 5
COMMON EMITTER TRANSISTOR CIRCUIT



```
C     LEONARD DAMKNER-BENDIX RADIO DIVISION-TOWSON,MD.
*RCO
CIRCUIT I
      DC ANALYSIS
B1    N*0,1<,R#,1,E#13.7
B2    N*1,2<,R#580.
B3    N*2,0<,R#2300.
B4    N*1,3<,RF180,%102,,198,<
B5    N*4,0<,R#1000,%900,,1100,<
B6    VR4,0<,R#54,%51,,62,<
B7    N*4,0<, R#1000,%900,,1100,<
B8    N*2,7<, R#60000,%12000,,748000,<
B9    N*5,9<, R#60000,%12000,,748000,<
B10   N*1,9<,R#580.
B11   N*9,9<,R#2300.
B12   N*3,2<,R#480,,E#-,6
B13   N*3,4<,R#2500.
B14   N*3,9<, R#480,,E#-,6
B15   N*3,9<, R#2500.
B16   N*4,6<,R#500,,E#-,6
B17   N*5,4<,R#1470.
B18   N*9,6<, R#500,,E#-,6
B19   N*7,5<, R#1470.
B20   N*1,5<,R#7,%,1<
B21   N*1,7<,R#76,%,1<
T1    N*19,13<,BETA#150,%100,,200,<
T2    R*16,15<,BETA#150,%100,,200,<
T3    R*16,17<,BETA#200,%150,,250,<
T4    R*18,19<,BETA#200,%150,,250,<
      SENSITIVITIES
      WORST CASE
      STANDARD DEVIATION
      PRINT,VOLTAGES,SENSITIVITIES,WORST CASE,STANDARD DEVIATION,CURRENT
      EXECUTE
```

NODE VOLTAGES

NODES	VOLTAGES			
1- 4	0.131894050 02	0.105488340 02	0.111452430 02	0.539817100 01
5- 8	0.120130840 02	0.470269970 01	0.109130840 02	0.539817100 01
9- 9	0.105488340 02			

ELEMENT CURRENTS

BRANCHES	CURRENTS			
1- 4	0.193950540 00	0.455305280-02	0.458644940-02	0.112464460-01
5- 8	0.539817060-07	0.439760609-01	0.539817060-07	0.787867370-06
9- 12	0.787867370-06	0.455305280-02	0.458644940-02	0.341844100-04
13- 16	0.558903060-02	0.341844100-04	0.558904040-02	0.190922470-03
17- 20	0.417970620-01	0.190922470-03	0.417970620-01	0.417963200-01
21- 21	0.417963200-01			

PARTIAL DERIVATIVES AND SENSITIVITIES OF NODE VOLTAGES

WITH RESPECT TO RESISTANCES

BRANCH	NODE	PARTIALS	SENSITIVITIES
1	1	-0.103837130 00	-0.193837070E-03
1	2	-0.830833910-01	-0.830833450E-04
1	3	-0.832577900-01	-0.832577350E-04
1	4	-0.544829030-01	-0.544819430E-04
1	5	-0.678321610-01	-0.678321230E-04
1	6	-0.533222490-01	-0.533222190E-04
1	7	-0.678321610-01	-0.678321230E-04
1	8	-0.544820030-01	-0.544819430E-04
1	9	-0.830833910-01	-0.830833450E-04
2	1	-0.894587990-05	-0.518860760E-06
2	2	-0.226803140-02	-0.131545890E-01
2	3	-0.179190460-02	-0.103930420E-01
2	4	0.512313340-01	0.297141610E 00
2	5	-0.134833120 01	-0.782031820E 01
2	6	0.458228150-02	0.265223200E-01
2	7	0.134214250 01	0.778442380E 01
2	8	-0.418566260-01	-0.242768350E 00
2	9	-0.135055830-02	-0.783323500E-02
3	1	0.247145730-05	0.568480750E-04
3	2	0.557499750-03	0.128547160E-01
3	3	0.455345660-03	0.104729420E-01
3	4	-0.130134600-01	-0.299311950E 00
3	5	0.342508000 00	0.787747790E 01
3	6	-0.116150350-02	-0.267144930E-01
3	7	-0.340935650 00	-0.784151460E 01
3	8	0.106356730-01	0.244591130E 00
3	9	0.343232090-03	0.789435210E-02
4	1	0.553924450-04	0.101506350E-03
4	2	-0.430565440-04	-0.787501740E-04
4	3	-0.147995220-03	-0.266729780E-03
4	4	-0.290352320-01	-0.522633960E-01
4	5	0.191674400-01	0.345013740E-01
4	6	-0.281257370-01	-0.506863050E-01
4	7	0.191674400-01	0.345013740E-01
4	8	-0.290352320-01	-0.522633960E-01
4	9	-0.430565440-04	-0.787501740E-04
5	1	-0.444969190-05	-0.444969050E-04
5	2	0.270150480-06	0.270150330E-05
5	3	-0.343770510-05	-0.343770370E-04
5	4	0.300397720-07	0.300397650E-01
5	5	-0.142577400-01	-0.142577350 00
5	6	0.250455590-02	0.250455520E-01
5	7	0.108669500-01	0.108669470 00
5	8	0.213469960-02	0.213469900E-01
5	9	-0.443622690-05	-0.443622540E-04
6	1	0.139494230-03	0.781154740E-04
6	2	0.146271290-03	0.819119060E-04
6	3	0.116069880-03	0.649790650E-04
6	4	0.362151350-02	0.202804710E-02
6	5	0.931441530-01	0.521460710E-01
6	6	0.544644040-02	0.304998410E-02
6	7	0.353141530-01	0.197707190E-01
6	8	0.362151350-02	0.202804710E-02
6	9	0.142721290-03	0.839319060E-04
7	1	-0.444969190-05	-0.444969050E-04
7	2	-0.443622690-05	-0.443622540E-04
7	3	-0.343770510-05	-0.343770370E-04
7	4	0.213469960-02	0.213469900E-01
7	5	0.108665500-01	0.108665470 00
7	6	0.250455590-02	0.250455520E-01
7	7	-0.142577400-01	-0.142577350 00
7	8	0.300337720-02	0.300337650E-01
7	9	0.270150480-06	0.270150330E-05
8	1	0.139041030-11	0.859716820E-08
8	2	0.134711790-09	0.227603950E-05
8	3	0.264445420-09	0.179823860E-05
8	4	-0.756077600-08	-0.514132700E-04
8	5	0.198982630-06	0.135309130E-02
8	6	-0.479053020-07	-0.459035980E-05
8	7	-0.198157090-06	-0.134746780E-02
8	8	0.617709190-08	0.420036090E-04
8	9	0.199312240-09	0.135532260E-05
9	1	0.139041030-11	0.859716820E-08
9	2	0.199312240-09	0.135532260E-05

9	3	0.266445420-09	0.179822846E-05
9	4	0.617700190-09	0.420036096-04
9	5	-0.198157950-04	-0.134746786-02
9	6	-0.675053020-09	-0.459035986-05
9	7	0.198982430-04	0.135308186-02
9	8	-0.756077690-08	-0.514133706-04
9	9	0.334711790-09	0.227603946-05
10	1	-0.894597990-05	-0.518866756-04
10	2	-0.139055830-02	-0.783323506-02
10	3	-0.179170460-02	-0.103936626-01
10	4	-0.418566260-01	-0.242768356-00
10	5	0.134216350-01	0.778663796-01
10	6	0.457281560-02	0.265223206-01
10	7	-0.136983120-01	-0.782031826-01
10	8	0.512133360-01	0.297161616-00
10	9	-0.226805360-02	-0.131656986-01
11	1	0.267165730-05	0.566867756-04
11	2	0.343232900-03	0.749435216-02
11	3	0.455346460-03	0.106729626-01
11	4	0.100326730-01	0.244551356-00
11	5	-0.360936450-00	-0.784151466-01
11	6	-0.114159090-02	-0.747164936-01
11	7	0.362580800-00	0.787767706-01
11	8	-0.139134600-01	-0.799318556-00
11	9	0.574293290-03	0.132547366-01
12	1	0.856653070-07	0.411709746-06
12	2	-0.128461140-04	-0.616421256-04
12	3	0.168880820-06	0.809379406-04
12	4	-0.681970010-03	-0.231346096-02
12	5	0.126847460-01	0.608869476-01
12	6	-0.432191690-04	-0.206491976-03
12	7	-0.196765460-01	-0.869374076-01
12	8	0.393777510-03	0.187011176-02
12	9	0.127069780-06	0.609831576-04
13	1	0.798613170-09	0.998015826-07
13	2	0.265380680-07	0.331779736-05
13	3	0.113836900-06	0.142799156-04
13	4	-0.338547110-05	-0.423183566-03
13	5	0.861665610-04	0.107784816-01
13	6	-0.479426270-06	-0.599467656-04
13	7	-0.856204430-04	-0.107025506-01
13	8	0.265820380-05	0.319775196-03
13	9	0.860478100-07	0.107584736-04
14	1	0.956845070-07	0.411709746-06
14	2	0.127069780-06	0.609831576-04
14	3	0.184869840-04	0.889239096-04
14	4	0.393777510-03	0.189013176-02
14	5	-0.135494460-01	-0.650907476-01
14	6	-0.430131490-04	-0.206491976-03
14	7	0.179987660-01	0.808869476-01
14	8	-0.481597110-03	-0.231346096-02
14	9	-0.128961140-04	-0.616421256-04
15	1	0.798613170-09	0.998015826-07
15	2	0.860678100-07	0.107584736-04
15	3	0.113836900-06	0.142799156-04
15	4	0.265820380-05	0.319775196-03
15	5	-0.856204430-04	-0.107025506-01
15	6	-0.403924770-04	-0.504407556-04
15	7	0.861665610-04	0.107784816-01
15	8	-0.338547110-05	-0.423183566-03
15	9	0.265820380-07	0.331725736-05
16	1	0.157778750-06	0.749545566-04
16	2	0.142717720-08	0.713563256-08
16	3	0.168632860-06	0.843464146-04
16	4	0.832758780-06	0.416379196-03
16	5	0.539093230-03	0.269965066-02
16	6	-0.883107810-06	-0.446798416-03
16	7	-0.419743370-03	-0.209881426-02
16	8	-0.744622150-06	-0.372210986-03
16	9	0.333991930-06	0.166995876-05
17	1	0.163646420-07	0.249295616-06
17	2	0.569029290-09	0.284135676-08
17	3	0.144752290-07	0.212743776-06
17	4	-0.417819960-05	-0.604874516-04
17	5	0.502643990-04	0.738886426-03
17	6	-0.451912920-05	-0.176330906-03
17	7	-0.378073470-04	-0.555747916-03
17	8	-0.772495980-05	-0.106396896-03
17	9	0.311656320-07	0.458134716-06
18	1	0.157778750-06	0.788963756-04
18	2	0.333991930-06	0.166995876-05
18	3	0.168689860-06	0.843464146-04
18	4	-0.744622150-04	-0.372210986-03
18	5	-0.410763170-03	-0.390981476-02
18	6	-0.888107810-06	-0.444398416-03
18	7	0.553001020-03	0.276865796-02
18	8	0.832758790-06	0.416379196-03
18	9	0.142717730-08	0.713563256-08
19	1	0.163646420-07	0.249295616-06
19	2	0.111656320-07	0.458134716-06
19	3	0.144752290-07	0.212743776-06
19	4	-0.772495980-05	-0.106396896-03
19	5	-0.378073470-04	-0.555747916-03
19	6	-0.451912920-05	-0.176330906-03
19	7	0.502643990-04	0.738886426-03
19	8	-0.417819960-05	-0.604874516-04
19	9	0.445892920-03	0.249915676-02
20	1	0.119069180-06	0.596975636-07
20	2	-0.105660340-04	-0.903013526-05
20	3	-0.138413480-04	-0.105344936-04
20	4	-0.375645630-03	-0.285345366-03
20	5	-0.307075330-01	-0.231339276-01
20	6	-0.627520170-04	-0.476027996-04
20	7	-0.109934460-01	-0.835501786-02
20	8	0.175975200-03	0.261306686-02
20	9	-0.173854490-04	-0.132758596-04
21	1	0.119069180-06	0.596975636-07
21	2	-0.173954690-04	-0.132758596-04
21	3	-0.138613480-04	-0.105346236-04
21	4	0.317597520-03	0.241306686-02
21	5	-0.109934460-01	-0.835501786-02
21	6	-0.627520170-04	-0.476027996-04
21	7	-0.307075330-01	-0.233399276-01
21	8	-0.375645630-03	-0.285345366-03
21	9	-0.105660340-04	-0.803013526-05

WITH RESPECT TO BETAS

BETA	NODE	PARTIALS	SENSITIVITIES
1	1	-0.739463590-06	-0.110919636-05
1	2	-0.246786720-04	-0.369879766-04
1	3	-0.105430080-03	-0.158145016-03
1	4	0.313551020-02	0.470326106-02
1	5	-0.798065890-01	-0.119706756-00
1	6	0.373732540-03	0.560598456-03
1	7	0.792987920-01	0.118964806-00
1	8	-0.233693280-02	-0.350539806-02
1	9	-0.797131280-04	-0.119569616-03
2	1	-0.739463590-06	-0.110919636-05
2	2	-0.797131280-04	-0.119569616-03
2	3	-0.105430080-03	-0.158145016-03
2	4	-0.233693280-02	-0.350539806-02
2	5	0.792987920-01	0.118964806-00
2	6	0.373732540-03	0.560598456-03
2	7	-0.798065890-01	-0.119706756-00
2	8	0.313551020-02	0.470326106-02
2	9	-0.246786720-04	-0.369879766-04
3	1	-0.126896480-05	-0.253997686-05
3	2	-0.501820120-07	-0.100361886-06
3	3	-0.112459920-05	-0.224911986-05
3	4	0.320733890-03	0.641467796-03
3	5	-0.390502340-02	-0.781003766-02
3	6	0.661864580-03	0.132369186-02
3	7	0.397229840-02	0.587446996-02
3	8	0.561304470-03	0.112269786-02

3	9	-0.242124690-05	-0.48424990E-05
4	1	-0.126998480-05	-0.25399248E-05
4	2	-0.243124690-05	-0.48424990E-05
4	3	-0.112659920-05	-0.22691158E-05
4	4	0.561304573-03	0.11226078E-02
4	5	0.293723940-02	0.58744498E-02
4	6	0.661846580-03	0.13236918E-02
4	7	-0.390509340-02	-0.78100376E-02
4	8	0.330733890-03	0.66146705E-03
4	9	-0.501830120-07	-0.10036398E-06

WITH RESPECT TO VOLTAGE SOURCES

BRANCH	NODE	PARTIALS	SENSITIVITIES
1	1	0.998909680 00	0.131896092E 00
1	2	0.799259040 00	0.105502192E 00
1	3	0.800037110 00	0.105723692E 00
1	4	0.524119020 00	0.691931118E-01
1	5	0.453543900 00	0.598136795E-01
1	6	0.512958230 00	0.677176498E-01
1	7	0.453543900 00	0.598136526E-01
1	8	0.524119020 00	0.691831118E-01
1	9	0.799259040 00	0.105502192E 00
12	1	-0.250639070-02	-0.150323378E-04
12	2	0.175780960 00	0.237617376E-03
12	3	-0.493177630 00	-0.295304642E-02
12	4	0.149091660 02	0.893947985E-01
12	5	-0.371068800 02	-0.222641095E 01
12	6	0.125844600 01	0.755046058E-02
12	7	0.369365470 02	0.221617228E 01
12	8	-0.115192150 02	-0.690115227E-01
12	9	-0.371716440 00	-0.223027975E-02
14	1	-0.250639070-02	-0.150323378E-04
14	2	-0.371716440 00	-0.223027975E-02
14	3	-0.493177630 00	-0.295304642E-02
14	4	-0.115192150 02	-0.690115227E-01
14	5	0.369365470 02	0.221617228E 01
14	6	0.125844600 01	0.755046058E-02
14	7	-0.371068800 02	-0.222641098E 01
14	8	0.149091660 02	0.893947985E-01
14	9	0.175780960 00	0.105417377E-02
16	1	-0.826034950-03	-0.495429058E-05
16	2	-0.747480480-05	-0.448493538E-07
16	3	-0.881566620-03	-0.530170468E-05
16	4	-0.436174090 00	-0.261705558E-02
16	5	-0.228200540 01	-0.136980238E-01
16	6	0.449213480 00	0.269527898E-02
16	7	0.219863660 01	0.131916198E-01
16	8	0.389907760 00	0.233864658E-02
16	9	-0.174035750-02	-0.104421438E-04
18	1	-0.826034950-03	-0.495429058E-05
18	2	-0.174035750-02	-0.104421438E-04
18	3	-0.881566620-03	-0.530170468E-05
18	4	0.389907760 00	0.233864658E-02
18	5	0.219863660 01	0.131916198E-01
18	6	0.449213480 00	0.269527898E-02
18	7	-0.283900560 01	-0.170340338E-01
18	8	-0.436174090 00	-0.261705558E-02
18	9	-0.747480480-05	-0.448493538E-07

STANDARD DEVIATIONS OF NODE VOLTAGES

NODE	STD. DEV.	NOM.-STD. DEV.	NOMINAL	NOM.+STD. DEV.
1	0.47443178E-03	0.13189130E 02	0.13189605D 02	0.13190079E 02
2	0.14686771E-02	0.10547365E 02	0.10548834D 02	0.10550303E 02
3	0.26583825E-02	0.11162583E 02	0.11165243D 02	0.11167900E 02
4	0.22336388E 00	0.51747570E 01	0.53981210D 01	0.56214848E 01
5	0.19771538E 01	0.80359297E 01	0.10013084D 02	0.11990237E 02
6	0.70800710E 00	0.44946518E 01	0.47026597D 01	0.49106665E 01
7	0.19771538E 01	0.80359297E 01	0.10013084D 02	0.11990237E 02
8	0.22336388E 00	0.51747570E 01	0.53981210D 01	0.56214848E 01
9	0.14686766E-02	0.10547365E 02	0.10548834D 02	0.10550303E 02

WORST CASE SOLUTIONS FOR NODE VOLTAGES

NODE	WCMIN	NOMINAL	WCMAX
1	PARTIAL W.R.T. R 8 HAS CHANGED SIGN AT MAX		
1	PARTIAL W.R.T. R 9 HAS CHANGED SIGN AT MAX		
1	0.131863088 02	0.131896048 02	0.131921310 02
2	PARTIAL W.R.T. R 5 HAS CHANGED SIGN AT MIN		
2	PARTIAL W.R.T. R 9 HAS CHANGED SIGN AT MIN		
2	PARTIAL W.R.T. R 8 HAS CHANGED SIGN AT MAX		
2	PARTIAL W.R.T. R 16 HAS CHANGED SIGN AT MAX		
2	PARTIAL W.R.T. R 17 HAS CHANGED SIGN AT MAX		
2	PARTIAL W.R.T. R 18 HAS CHANGED SIGN AT MAX		
2	PARTIAL W.R.T. T 3 HAS CHANGED SIGN AT MAX		
2	PARTIAL W.R.T. T 4 HAS CHANGED SIGN AT MAX		
2	PARTIAL W.R.T. R 16 HAS CHANGED SIGN AT MAX		
2	0.105427648 02	0.105488348 02	0.105605250 02
3	PARTIAL W.R.T. R 8 HAS CHANGED SIGN AT MAX		
3	PARTIAL W.R.T. R 9 HAS CHANGED SIGN AT MAX		
3	0.111537428 02	0.111652428 02	0.111874658 02
4	PARTIAL W.R.T. R 9 HAS CHANGED SIGN AT MIN		

```
 4 PARTIAL W.R.T.  R 16  HAS CHANGED SIGN AT MIN
 4 PARTIAL W.R.T.  R 20  HAS CHANGED SIGN AT MIN
 4 PARTIAL W.R.T.  T  3  HAS CHANGED SIGN AT MIN
 4 PARTIAL W.R.T.  R  8  HAS CHANGED SIGN AT MAX
 4 PARTIAL W.R.T.  R 17  HAS CHANGED SIGN AT MAX
 4 PARTIAL W.R.T.  R 18  HAS CHANGED SIGN AT MAX
 4 PARTIAL W.R.T.  R 21  HAS CHANGED SIGN AT MAX
 4 PARTIAL W.R.T.  T  4  HAS CHANGED SIGN AT MAX
 4       0.41004992E 01      0.53981209E 01      0.69077367D 01
 5 PARTIAL W.R.T.  R  1  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  R  8  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  R 17  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  R 18  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  R 21  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  T  4  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  E  1  HAS CHANGED SIGN AT MIN
 5 PARTIAL W.R.T.  R  4  HAS CHANGED SIGN AT MAX
 5 PARTIAL W.R.T.  R  9  HAS CHANGED SIGN AT MAX
 5 PARTIAL W.R.T.  R 16  HAS CHANGED SIGN AT MAX
 5 PARTIAL W.R.T.  R 19  HAS CHANGED SIGN AT MAX
 5 PARTIAL W.R.T.  R 20  HAS CHANGED SIGN AT MAX
 5 PARTIAL W.R.T.  T  3  HAS CHANGED SIGN AT MAX
 5      -0.45483303E 01      0.10013083E 02      0.20156366D 02
 6       0.36474390E 01      0.47026596E 01      0.59658662D 01
 7 PARTIAL W.R.T.  R  1  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  R  9  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  R 16  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  R 19  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  R 20  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  T  3  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  E  1  HAS CHANGED SIGN AT MIN
 7 PARTIAL W.R.T.  R  4  HAS CHANGED SIGN AT MAX
 7 PARTIAL W.R.T.  R  8  HAS CHANGED SIGN AT MAX
 7 PARTIAL W.R.T.  R 17  HAS CHANGED SIGN AT MAX
 7 PARTIAL W.R.T.  R 18  HAS CHANGED SIGN AT MAX
 7 PARTIAL W.R.T.  R 21  HAS CHANGED SIGN AT MAX
 7 PARTIAL W.R.T.  T  4  HAS CHANGED SIGN AT MAX
 7      -0.45483303E 01      0.10013083E 02      0.20156366D 02
 8 PARTIAL W.R.T.  R  8  HAS CHANGED SIGN AT MIN
 8 PARTIAL W.R.T.  R 18  HAS CHANGED SIGN AT MIN
 8 PARTIAL W.R.T.  R 21  HAS CHANGED SIGN AT MIN
 8 PARTIAL W.R.T.  T  4  HAS CHANGED SIGN AT MIN
 8 PARTIAL W.R.T.  R  9  HAS CHANGED SIGN AT MAX
 8 PARTIAL W.R.T.  R 16  HAS CHANGED SIGN AT MAX
 8 PARTIAL W.R.T.  R 19  HAS CHANGED SIGN AT MAX
 8 PARTIAL W.R.T.  R 20  HAS CHANGED SIGN AT MAX
 8 PARTIAL W.R.T.  E  1  HAS CHANGED SIGN AT MAX
 8       0.41004992E 01      0.53981209E 01      0.69077367D 01
 9 PARTIAL W.R.T.  R  7  HAS CHANGED SIGN AT MIN
 9 PARTIAL W.R.T.  R  8  HAS CHANGED SIGN AT MIN
 9 PARTIAL W.R.T.  R  9  HAS CHANGED SIGN AT MAX
 9 PARTIAL W.R.T.  R 16  HAS CHANGED SIGN AT MAX
 9 PARTIAL W.R.T.  R 18  HAS CHANGED SIGN AT MAX
 9 PARTIAL W.R.T.  R 19  HAS CHANGED SIGN AT MAX
 9 PARTIAL W.R.T.  T  3  HAS CHANGED SIGN AT MAX
 9 PARTIAL W.R.T.  T  4  HAS CHANGED SIGN AT MAX
 9 PARTIAL W.R.T.  E 18  HAS CHANGED SIGN AT MAX
 9       0.10547764E 02      0.10568834E 02      0.10580959D 02
```

```
*EBCDIC
CIRCUIT 2
C   LOW PASS FILTER
      AC ANALYSIS
B1    N(0,1),R=.0000001,E=1.
B2    N(1,2),R=1.
B3    N(2,0),C=.15923E-6
      FREQUENCY=1000000.
      PRINT,NV,CA,BP,MI
      MODIFY
      FREQUENCY=100.E3(2)6.4E6
      EXECUTE
```

ROW NODE	COL NODE	NODAL ADMITTANCE MATRIX	
REAL 1	1- 2	0.10000016D 07	-0.10000000D 01
IMAG		0.0	0.0
REAL 2	1- 2	-0.10000000D 01	0.10000000D 01
IMAG		0.0	0.10004702D 01

NODES	EQUIVALENT CURRENT VECTOR	
REAL 1- 2	0.10000060 07	0.0
IMAG	0.0	0.0

FREQ = 0.99999994E 06

NODES	NODE VOLTAGES	
MAG 1- 2	0.99999946E 00	0.76969176E 00
PHA	-0.28647846E-04	-0.45013489E 02

BRANCHES	ELEMENT CURRENTS		
MAG 1- 3	0.70727259E 00	0.70727259E 00	0.70727259E 00
PHA	0.44986496E 02	0.44986496E 02	0.44986496E 02

BRANCHES	BRANCH POWER	
MAG 1- 3	0.50023418E-06	0.50023448E 00	0.97144515E-16

ROW NODE	COL NODE	NODAL ADMITTANCE MATRIX	
REAL 1	1- 2	0.10000016D 07	-0.10000000D 01
IMAG		0.0	0.0
REAL 2	1- 2	-0.10000000D 01	0.10000000D 01
IMAG		0.0	0.10004702D 01

NODES	EQUIVALENT CURRENT VECTOR	
REAL 1- 2	0.10000060 07	0.0
IMAG	0.0	0.0

FREQ = 0.99999994E 06

NODES	NODE VOLTAGES	
MAG 1- 2	0.99999946E 00	0.99553240E 00
PHA	-0.54754549E-05	-0.57132445E 01

BRANCHES	ELEMENT CURRENTS		
MAG 1- 3	0.99550768E-01	0.99550768E-01	0.99550768E-01
PHA	0.94286713E 02	0.94286713E 02	0.94286713E 02

BRANCHES	BRANCH POWER	
MAG 1- 3	0.99102295E-08	0.99102147E-02	-0.43368997E-17

ROW NODE	COL NODE	NODAL ADMITTANCE MATRIX	
REAL 1	1- 2	0.10000016D 07	-0.10000000D 01
IMAG		0.0	0.0
REAL 2	1- 2	-0.10000000D 01	0.10000000D 01
IMAG		0.0	0.10004702D 01

NODES	EQUIVALENT CURRENT VECTOR	
REAL 1- 2	0.10000060 07	0.0
IMAG	0.0	0.0

FREQ = 0.12999981E 06

NODES	NODE VOLTAGES	
MAG 1- 2	0.99999946E 00	0.98055787E 00
PHA	-0.11023194E-04	-0.11315124E 02

BRANCHES	ELEMENT CURRENTS		
MAG 1- 3	0.19620478E 00	0.19620478E 00	0.19620478E 00
PHA	0.78684860E 02	0.78684860E 02	0.78684860E 02

```
   BRANCHES          BRANCH POWER

MAG    1- 3  0.38496299E-07  0.38496319E-01 -0.60715322E-17

   ROW   COL
   NODE  NODE          NODAL ADMITTANCE MATRIX

REAL  1     1- 2  0.10000016D 07 -0.10000000D 01
IMAG              0.0              0.0
REAL  2     1- 2 -0.10000000D 01  0.10000000D 01
IMAG              0.0              0.40018833D 00

   NODES          EQUIVALENT CURRENT VECTOR

REAL  1- 2  0.10000006D 07  0.0
IMAG        0.0             0.0

FREQ =  0.39999987E 06

   NODES          NODE VOLTAGES

MAG    1- 2  0.99999987E 00  0.92941625E 00
PHA         -0.19763873E-04 -0.21810730E 02

   BRANCHES          ELEMENT CURRENTS

MAG    1- 3  0.37154132E 00  0.37154132E 00  0.37154132E 00
PHA          0.68189209E 02  0.68189209E 02  0.68189209E 02

   BRANCHES          BRANCH POWER

MAG    1- 3  0.13804288E-06  0.13804293E 00  0.0

   ROW   COL
   NODE  NODE          NODAL ADMITTANCE MATRIX

REAL  1     1- 2  0.10000016D 07 -0.10000000D 01
IMAG              0.0              0.0
REAL  2     1- 2 -0.10000000D 01  0.10000000D 01
IMAG              0.0              0.80037683D 00

   NODES          EQUIVALENT CURRENT VECTOR

REAL  1- 2  0.10000006D 07  0.0
IMAG        0.0             0.0

FREQ =  0.79999994E 06

   NODES          NODE VOLTAGES

MAG    1- 2  0.99999958E 00  0.78072494E 00
PHA         -0.27952017E-04 -0.38672998E 02

   BRANCHES          ELEMENT CURRENTS

MAG    1- 3  0.62487417E 00  0.62487417E 00  0.62487417E 00
PHA          0.51326996E 02  0.51326996E 02  0.51326996E 02

   BRANCHES          BRANCH POWER

MAG    1- 3  0.39046751E-06  0.39046770E 00  0.13877788E-16

   ROW   COL
   NODE  NODE          NODAL ADMITTANCE MATRIX

REAL  1     1- 2  0.10000016D 07 -0.10000000D 01
IMAG              0.0              0.0
REAL  2     1- 2 -0.10000000D 01  0.10000000D 01
IMAG              0.0              0.16007528D 01

   NODES          EQUIVALENT CURRENT VECTOR

REAL  1- 2  0.10000006D 07  0.0
IMAG        0.0             0.0

FREQ =  0.15999990E 07

   NODES          NODE VOLTAGES

MAG    1- 2  0.99999923E 00  0.52981925E 00
PHA         -0.25745554E-04 -0.58006714E 02

   BRANCHES          ELEMENT CURRENTS

MAG    1- 3  0.84810966E 00  0.84810966E 00  0.84810966E 00
PHA          0.31993240E 02  0.31993240E 02  0.31993240E 02

   BRANCHES          BRANCH POWER

MAG    1- 3  0.71929963E-06  0.71929008E 00  0.55511151E-16

   ROW   COL
   NODE  NODE          NODAL ADMITTANCE MATRIX
```

```
REAL  1    1- 2  0.100000160 07 -0.100000000 01
IMAG             0.0              0.0
REAL  2    1- 2 -0.100000000 01  0.100000000 01
IMAG             0.0              0.32015047D 01

      NODES              EQUIVALENT CURRENT VECTOR

REAL  1- 2  0.100000060 07  0.0
IMAG        0.0             0.0

FREQ =  0.31999970E 07

      NODES              NODE VOLTAGES

MAG   1- 2  0.99999905E 00  0.29814696E 00
PHA        -0.16305625E-04 -0.72653641E 02

      BRANCHES           ELEMENT CURRENTS

MAG   1- 3  0.95451897E 00  0.95451897E 00  0.95451897E 00
PHA         0.17346329E 02  0.17346329E 02  0.17346329E 02

      BRANCHES           BRANCH POWER

MAG   1- 3  0.91110599E-06  0.91110659E 00  0.41433343E-16

      ROW  COL
      NODE NODE          NODAL ADMITTANCE MATRIX

REAL  1    1- 2  0.100000160 07 -0.100000000 01
IMAG             0.0              0.0
REAL  2    1- 2 -0.100000000 01  0.100000000 01
IMAG             0.0              0.64030133D 01

      NODES              EQUIVALENT CURRENT VECTOR

REAL  1- 2  0.100000060 07  0.0
IMAG        0.0             0.0

FREQ =  0.63999980E 07

      NODES              NODE VOLTAGES

MAG   1- 2  0.99999999E 00  0.15430576E 00
PHA        -0.87351755E-05 -0.81123413E 02

      BRANCHES           ELEMENT CURRENTS

MAG   1- 3  0.98802209E 00  0.98802209E 00  0.98802209E 00
PHA         0.98765373E 01  0.98765373E 01  0.98765373E 01

      BRANCHES           BRANCH POWER

MAG   1- 3  0.97615704E-05  0.97618771E 00 -0.26020852E-17
```

```
CIRCUIT 3
      TRANSIENT ANALYSIS
B1    N(1,0),R=.01,E=-2.5
B2    N(1,2),C=.01E-06,E0=-2.5
B3    N(1,2),R=1.E+14
B4    N(2,0),R=1.E+14,E=-2.2
      TIME STEP=21600.
      OUTPUT INTERVAL=1
      FINISH TIME=21600.
      PRINT VOLTAGES,CURRENTS
      EXECUTE

   T =   0.0

NODES              NODE VOLTAGES

  1- 2      0.249999816 01    0.38146815E-07

BRANCHES           ELEMENT CURRENTS

  1- 4     -0.222044640-12    0.568436190-13    0.249999900-13   -0.219999970-13

   T =   0.21600000E 05

NODES              NODE VOLTAGES

  1- 2      0.24999990E 01    0.97315967E-01

BRANCHES           ELEMENT CURRENTS

  1- 4      0.0               -0.450536680-13   0.240268300-13   -0.210268380-13
```

```
/CIRCUIT 4
C  TRANSISTOR SWEEP CIRCUIT
     TRANSIENT ANALYSIS
B1   N(0,2),R=1000.
E1   P(5),-2,2,2,2,-2,-2,-2,-2
B2   N(1,2),R=107000.
B3   N(1,3),R=1000.
B4   N(3,0),C=.15E-06
B5   N(1,0),R=.01,E=15.
B6   N(4,2),R=(.1,10.E6)
B7   N(0,4),R=6R0.,E=-.6
B8   N(0,3),R=(12500.,30.E6)
B9   N(3,5),R=30.E6
B10  N(5,0),R=1.E6
B11  N(0,7),R=(30.E6,1.E6)
S1   R=6.(7.8.61.ON
S2   R=9.(7.8.111.OFF
T1   R(7,8),BETA=(150.0)
T2   R(7,10),BETA=150
     SHORT=.1
     OPEN=100.E6
     TIME STEP=5.E-06
     OUTPUT INTERVAL=5
     FINISH TIME=275.E-06
     PRINT,VOLTAGES,CURRENTS
     EXECUTE


  T =  0.0

NODES                NODE VOLTAGES
  1- 4    -0.149998486E 02   -0.109673658E 01   0.149335837E-01   -0.109616358E 01
  5- 5    -0.246081276 00

BRANCHES             ELEMENT CURRENTS
  1- 4    -0.903733360-03   -0.129940010-03   -0.150138510-01   0.149335870 00
  5- 8     0.151437910-01    0.103367340-02    0.103367340-02   0.159049860 00
  9- 11    0.873616200-08    0.139409000-06   -0.446786100-09

SWITCH  2 IS ON

  T =  0.0

NODES                NODE VOLTAGES
  1- 4    -0.149998486E 02   -0.109691216E 01   0.160543066E-01   -0.109590826E 01
  5- 5    -0.246084077E 00

BRANCHES             ELEMENT CURRENTS
  1- 4    -0.903201010-03   -0.129947390-03   -0.150159070-01   -0.253498580-01
  5- 8     0.151458440-01    0.103314360-02    0.103314340-02   -0.535143640-09
  9- 11    0.889027980-08    0.139447790-06   -0.160339460-01


  T =  0.8666991E-05

NODES                NODE VOLTAGES
  1- 4    -0.149998486E 02   -0.664756119E 00   -0.238491635E-01   -0.664755130E 00
  5- 5    -0.237755446E-01

BRANCHES             ELEMENT CURRENTS
  1- 4     0.135598191D-04   -0.134133570-03   -0.149759890-01   -0.792648240-04
  5- 8     0.151101320-01    0.990653300-04    0.990653300-04   0.704977180-09
  9- 11   -0.245409940-11    0.771914790-07    0.149957430-01

SWITCH  2 IS OFF

  T =  0.8666991E-05

NODES                NODE VOLTAGES
  1- 4    -0.149998468E 02   -0.664756119E 00   -0.238406078E-01   -0.664755130E 00
  5- 5    -0.237755446E-01

BRANCHES             ELEMENT CURRENTS
  1- 4     0.135598191D-04   -0.134133570-03   -0.149759870-01   -0.114357730-03
  5- 8     0.151101310-01    0.990653300-04    0.990653300-04   0.149617760-01
  9- 11   -0.245409940-11    0.771914790-07    0.149957430-01

  T =  0.9546337E-05

NODES                NODE VOLTAGES
  1- 4    -0.149998469E 02   -0.600007578E 00   -0.114989934E 00   -0.600007578E 00
  5- 5    -0.367660946E-05

BRANCHES             ELEMENT CURRENTS
  1- 4     0.135621190-03   -0.134577960-03   -0.148848590-01   -0.148733590-01
  5- 8     0.150194370-01    0.157705070-07    0.157705070-07   0.115647260-04
  9- 11   -0.149389660-08    0.476934150-07    0.383209710-08

SWITCH  1 IS OFF

  T =  0.9586037E-05

NODES                NODE VOLTAGES
  1- 4    -0.149998469E 02   -0.600023156 00   -0.116477676E 00   -0.599999968 00
  5- 5     0.137672074E-06

BRANCHES             ELEMENT CURRENTS
  1- 4     0.135577810-03   -0.134577810-03   -0.148833710-01   -0.148833590-01
  5- 8     0.150170490-01    0.231926250-11    0.231926190-11   0.388259030-08
  9- 11   -0.388259390-08    0.676429760-07    0.388259030-08


  T =  0.2499999E-04
```

T = 0.2499999E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999864E 02	0.18431330E 01	-0.15479555E 01	-0.59988266E 00
5- 5	0.58664184E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.1576954B0-03	-0.15741118D-03	-0.13451907D-01	-0.13451752D-01
5- 8	0.1360931B0-01	-0.24430151D-06	-0.24430151D-06	0.51598540D-07
9- 11	-0.5180049B0-07	0.19911094D-07	0.51598540D-07	

T = 0.4999996E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999884E 02	0.18431330E 01	-0.35820427E 01	-0.59988266E 00
5- 5	0.58555699E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.1576955670-03	-0.15741137D-03	-0.11417843D-01	-0.11417487D-01
5- 8	0.1157525I0-01	-0.24430149D-06	-0.24430149D-06	0.11940144D-06
9- 11	-0.1194031B0-06	-0.47491804D-07	0.11940144D-06	

T = 0.7499993E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999901E 02	0.18431320E 01	-0.53085213E 01	-0.59988266E 00
5- 5	0.59463615E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.1576555830-03	-0.15741153D-03	-0.9691374G0-02	-0.9690R4870-02
5- 8	0.9848791I0-02	-0.24430147D-06	-0.24430147D-06	0.1769507I0-06
9- 11	-0.1769926B0-06	-0.10544106D-06	0.1769507I0-06	

T = 0.9041493E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999910E 02	-0.60019565E 00	-0.62436473E 01	-0.599999996E 00
5- 5	-0.32309024E-06			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.1345767D0-03	-0.13457677D-03	-0.875926260-02	-0.875863850-02
5- 8	0.889381940-02	0.195695850-10	0.195695840-10	0.208021590-06
9- 11	-0.208021580-06	-0.136494010-06	0.208021590-06	

SWITCH 1 IS ON

T = 0.9041493E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999910E 02	-0.60019565E 00	-0.62436473E 01	-0.599999996E 00
5- 5	-0.32309024E-06			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.1345767D0-03	-0.13457677D-03	-0.875926260-02	-0.875863850-02
5- 8	0.889381940-02	0.195695850-10	0.195695840-10	0.208021590-06
9- 11	-0.208021580-06	-0.136494010-06	0.208021590-06	

T = 0.9956531E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999850E 02	-0.10735254E 01	-0.22824488E 00	-0.10734262E 01
5- 5	-0.23671323E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.8961540D-03	-0.1301527D-03	-0.1477460D-01	0.1760039D 00
5- 8	0.1489215D-01	0.9863065D-03	0.9863065D-03	0.7508164I0-08
9- 11	-0.3772331D-10	0.1279247D-06	0.7928177D-08	

SWITCH 2 IS ON

T = 0.9956531E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999850E 02	-0.10735254E 01	-0.22824488E 00	-0.10734262E 01
5- 5	-0.23671323E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.8961540D-03	-0.1301527D-03	-0.1477460D-01	0.1760039D 00
5- 8	0.1489215D-01	0.9863065D-03	0.9863065D-03	0.7508164I0-08
9- 11	0.3822782D-09	0.1283447D-06	0.1407781D 00	

T = 0.9999999E-04

NODES	NODE VOLTAGES			
1- 4	-0.14999849E 02	-0.10960121E 01	-0.9549766RE-01	-0.10959091E 01
5- 5	-0.24795437E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.9012023D0-03	-0.1299437D0-09	-0.1490435D-01	0.4478173D0-01
5- 8	0.1503429D0-01	0.1033144I0-02	0.1033144I0-02	0.3183257D0-08
9- 11	0.5081889D0-08	0.1357244B0-06	0.5968609D0-01	

T = 0.1249999E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.10960121E 01	-0.23961436E-01	-0.10959091E 01
5- 5	-0.24795437E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.90320730D-03	-0.12994239D-03	-0.14975886D-01	0.89574947D-08
5- 8	0.15105829D-01	0.10331447D-02	0.10331447D-02	0.79871453D-09
9- 11	0.74664320D-08	0.13810903D-06	0.14975902D-01	

T = 0.1499998E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.10960121E 01	-0.23961421E-01	-0.10959091E 01
5- 5	-0.24795437E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.90320730D-03	-0.12994239D-03	-0.14975886D-01	0.17917172D-14
5- 8	0.15105829D-01	0.10331447D-02	0.10331447D-02	0.79871495D-09
9- 11	0.74664325D-08	0.13810903D-06	0.14975893D-01	

T = 0.1749999E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.10960121E 01	-0.23961421E-01	-0.10959091E 01
5- 5	-0.24795447E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.90320730D-03	-0.12994239D-03	-0.14975886D-01	0.0
5- 8	0.15105829D-01	0.10331447D-02	0.10331447D-02	0.79871405D-09
9- 11	0.74664325D-08	0.13810903D-06	0.14975893D-01	

T = 0.1836570E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.64806712E 00	-0.23961410E-01	-0.64805710E 00
5- 5	-0.24029454E-01			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.34009673D-04	-0.13612979D-03	-0.14975886D-01	0.79171705D-09
5- 8	0.15113015D-01	0.10011912D-03	0.10011912D-03	0.79871367D-09
9- 11	0.22348234D-11	0.77256667D-07	0.14975885D-01	

SWITCH 2 IS OFF

T = 0.1836570E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.64806712E 00	-0.23957079E-01	-0.64805710E 00
5- 5	-0.24029454E-01			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.34009673D-04	-0.13612979D-03	-0.14975891D-01	0.43915342D-04
5- 8	0.15113027D-01	0.10011912D-03	0.10011912D-03	0.15919789D-01
9- 11	0.22348234D-11	0.77256667D-07	0.79856761D-09	

T = 0.1749998E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.64806712E 00	-0.11448744E 00	-0.64980249E 00
5- 5	-0.24029454E-01			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.13497219D-03	-0.13657203D-03	-0.14983361D-01	-0.14907579D-01
5- 8	0.15117347D-01	-0.22315594D-06	-0.22315594D-06	-0.26196378D-04
9- 11	-0.13847044D-08	0.67429095D-07	0.38829159D-08	

SWITCH 1 IS OFF

T = 0.1749998E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.59947172E 00	-0.11797565E 00	-0.59999990E 00
5- 5	-0.11602509E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.13459113D-03	-0.13459113D-03	-0.14981873D-01	-0.14981873D-01
5- 8	0.15016454D-01	-0.32817839D-10	-0.32817839D-10	0.39325221D-08
9- 11	-0.39325240D-04	0.67493049D-07	0.39325221D-08	

T = 0.1999998E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999864E 02	0.18431387E 01	-0.15491142E 01	-0.59988266E 00
5- 5	0.59444256E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.15765553D-03	-0.15741123D-03	-0.13459748D-01	-0.13460599D-01
5- 8	0.13608159D-01	-0.24430207D-06	-0.24430207D-06	0.51637166D-07
9- 11	-0.51639121D-07	0.19872469D-07	0.51637166D-07	

T = 0.2249998E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999884E 02	0.18431387E 01	-0.35830259E 01	-0.59988266E 00
5- 5	0.59559786E-04			

NODES	NODE VOLTAGES			
1- 4	-0.14999884E 02	0.18431377E 01	-0.35830259E 01	-0.59988266E 00
5- 5	0.58565786E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.15765579D-03	-0.15741142D-03	-0.11416957D-01	-0.11416498D-01
5- 8	0.11579268D-01	-0.24430205D-06	-0.24430205D-06	0.11943423D-06
9- 11	-0.11943618D-06	-0.47924588D-07	0.11943423D-06	

T = 0.2499993E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999901E 02	0.18431377E 01	-0.53093557E 01	-0.59988266E 00
5- 5	0.58463702E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.15765588D-03	-0.15741158D-03	-0.96095446D-02	-0.96001388D-02
5- 8	0.98679566D-02	-0.24430296D-06	-0.24430204D-06	0.17697853D-06
9- 11	-0.17698048D-06	-0.10548890D-06	0.17697853D-06	

T = 0.7654134E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999910E 02	-0.60018998E 00	-0.67414017E 01	-0.59999996E 00
5- 5	-0.22298446E-06			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.13457681D-03	-0.13457683D-03	-0.87585081D-02	-0.87578840D-02
5- 8	0.88918869D-02	0.19002769D-10	0.19002769D-10	0.20804674D-06
9- 11	-0.20804674D-06	-0.13621169D-06	0.20804674D-06	

SWITCH 1 IS ON

T = 0.2654134E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999910E 02	-0.60006195E 00	-0.62472256E 01	-0.60006195E 00
5- 5	-0.31230215E-04			

BRANCHES	ELEMENT CURRENTS			
1- 4	0.13444881D-03	-0.13457802D-03	-0.87576843D-02	-0.87385792D-02
5- 8	0.88922623D-02	0.19215749D-06	0.19215749D-06	0.51876944D-03
9- 11	-0.20807316D-06	-0.13654020D-06	0.20807420D-06	

T = 0.2745637E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999850E 02	-0.10735416E 01	-0.23811632E 00	-0.10734434E 01
5- 5	-0.23672146E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.8561882D-03	-0.13015241D-03	-0.14761734D-01	0.13329827D 00
5- 8	0.14891886D-01	0.98634064D-03	0.98634064D-03	0.14797013D 00
9- 11	-0.4649774D-10	0.12701794D-06	0.79372125D-08	

SWITCH 2 IS ON

T = 0.2745637E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999850E 02	-0.10735416E 01	-0.22559005E 00	-0.10734434E 01
5- 5	-0.23672146E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.8561882D-03	-0.13015241D-03	-0.14774350D-01	0.13618924D 00
5- 8	0.14924502D-01	0.98634064D-03	0.98634064D-03	0.75106845D-08
9- 11	0.5749462D-09	0.12833489D-06	0.14093768D 00	

T = 0.2749984E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999849E 02	-0.10960436E 01	-0.95557451E-01	-0.10959406E 01
5- 5	-0.24797010E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.9032681D-03	-0.12994210D-03	-0.14974961D-01	0.44181935D-01
5- 8	0.15034233D-01	0.10331102D-02	0.10331102D-02	0.31952495D-08
9- 11	0.5088942D-08	0.13572476D-06	0.59723344D-01	

T = 0.2749982E-03

NODES	NODE VOLTAGES			
1- 4	-0.14999848E 02	-0.10960436E 01	-0.27235427E-01	-0.10959406E 01
5- 5	-0.24797010E 00			

BRANCHES	ELEMENT CURRENTS			
1- 4	-0.9032681D-03	-0.12994209D-03	-0.14972612D-01	0.20496651D-02
5- 8	0.15102554D-01	0.10331102D-02	0.10331102D-02	0.90785432D-09
9- 11	0.735781640-08	0.13800415D-06	0.17022274D-01	

```
CIRCUIT 5
C    COMMON EMITTER TRANSISTOR
*BCD
     DC ANALYSIS
B1   N%1,2<,R#330000.%,1<
B2   N%1,3<,R#1180.%,1<
B3   N%0,1<,R#.1%,1<,FW=13.2%,1<
B4   N%0,2<,R#680.%,1<,FW=.6%,1<
B5   N%0,3<,R#12500.%,1<
T1   B%4,5<,BETA#150%,1<
     SENSITIVITIES
     WORST CASE
     STANDARD DEVIATION
     PRINT,NV,CA,SP,SE,ST,WO
     EXECUTE

NODE VOLTAGES

NODES          VOLTAGES
  1- 3    -0.131993760 02   -0.618299660 00   -5.5894A7330 01

ELEMENT CURRENTS

BRANCHES          CURRENTS
  1- 4   -0.391244700-04   -0.619979560-02   -5.423377360-02   0.381749990-04
  4- 5    0.619024290-02

ELEMENT POWER LOSSES

BRANCHES          POWER LOSSES
  1- 4    0.479664890-03   0.452167410-01   0.388592610-05   0.497664060-04
  4- 5    0.344084970-01

PARTIAL DERIVATIVES AND SENSITIVITIES OF NODE VOLTAGES

WITH RESPECT TO RESISTANCES

BRANCH  NODE      PARTIALS          SENSITIVITIES
  1      1   -0.159240950-08   -0.525561065-05
  1      2   -0.551100090-07   -0.182229395-03
  1      3   -0.184582740-06   -0.615772795-01
  2      1   -0.452491630-07   -0.533899245-06
  2      2   -0.657199120-10   -0.775496785-09
  2      3    0.565429050-02    0.657440895-01
  3      1    0.423341180-07    0.473361075-05
  3      2    0.405341490-05    0.405363745-03
  3      3    0.266518770-02    0.266518675-05
  4      1   -0.159249980-08   -0.764199695-08
  4      2   -0.190689760-06   -0.182731055-03
  4      3   -0.185649590-04   -0.295533755-04
  5      1   -0.344712220-08   -0.437892175-06
  5      2   -0.500671320-11   -0.425483995-09
  5      3   -0.406795220-06   -0.509493565-02

WITH RESPECT TO BETAS

BETA    NODE      PARTIALS          SENSITIVITIES
  1      1   0.348341210-05    0.522551445E-05
  1      2   0.505942170-08    0.759317595E-08
  1      3   0.411077790-01    0.616616275E-01

WITH RESPECT TO VOLTAGE SOURCES

BRANCH  NODE      PARTIALS          SENSITIVITIES
  3      1   0.239950220-00    0.131993675 00
  3      2   0.145236140-02    0.191771665-03
  3      3   0.626335360-00    0.660122505-01
  4      1   0.417739500-04    0.751664155E-06
  4      2   0.999547630 00    0.999128015-02
  4      3   0.499404220-00    0.799364735-02

STANDARD DEVIATIONS OF NODE VOLTAGES

NODE    STD. DEV.          NOM.-STD. DEV.     NOMINAL          NOM.+STD. DEV.
  1   0.439977885 00   -0.136393565 02   -0.131993765 02   -0.125599985 02
  2   0.139997225-01   -0.638299355 00   -0.618299465 00   -0.598299095 00
  3   0.411229955 00   -0.630610285 01   -0.589487335 01   -0.548764435 01

WORST CASE SOLUTIONS FOR NODE VOLTAGES

NODE      WCMIN          NOMINAL          WCMAX
  1   -0.145199495E 02   -0.131993765 02   -0.118795575 02
  2   -0.684594995E 00   -0.618299665 00   -0.556347655 00
  3   -0.849435915E 01   -0.589487375 01   -0.339094745 01
```

PROGRAM DESIGN FOR SCIENTIFIC COMPUTING

By

DR. K. ERHARD WITTMER

Dr. Wittmer received his Dipl.-Ing. in Electrical Engineering from the Institut of Technology, Aachen, Germany in 1959. In 1962 Dr. Wittmer received his Dr.-Ing. degree in Electrical Engineer, also from the Institut at Aachen.

In 1959-1962 Dr. Wittmer was a Research Fellow at the Institut, and in 1962-1963 he was an Assistant Professor at the Department for the HF and UHF Techniques, also at the Institut.

Since 1963 Dr. Wittmer has been with Bell Telephone Laboratories, Incorporated.

At the Institut of Technology, Aachen, Dr. Wittmer was engaged in research in microwave techniques and network theory. At the Bell Telephone Laboratories he has been concerned with the development of computer programs for engineering design and analysis.

From 1965 he has been heading a group concerned with the application of computers in the engineering design process.

PROGRAM DESIGN FOR SCIENTIFIC COMPUTING

by

K. E. WITTMER
Bell Telephone Laboratories, Incorporated
North Andover, Massachusetts

ABSTRACT

The design of computer programs is a very essential part of computing and often it does not receive enough attention to guarantee economical computer usage.

Scientific computing is reviewed from the standpoint of the computer user.

Program design considerations are presented with emphasis on the development of application programs.

Important program design principles are demonstrated by means of a user-oriented, special-purpose network analysis program.

. . . . . .

Because of review complications, Dr. Wittmer was unable to submit his complete paper in time for publication.

11866

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

ELECTRONICS RESEARCH CENTER

575 TECHNOLOGY SQUARE

CAMBRIDGE, MASS. 02139